

GATED COMPETITIVE SYSTEMS FOR UNSUPERVISED SEGMENTATION
AND MODELING OF PIECEWISE STATIONARY SIGNALS

By

CRAIG LANGDALE FANCOURT

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1998

ACKNOWLEDGEMENTS

First and foremost I wish to thank my advisor, Dr. José Principe. He allowed me the freedom to explore, while at the same time provided invaluable insight without which this dissertation would not have been possible.

Next, I wish to thank my parents for their unending love and support and for instilling in me a love of learning.

I also wish to thank the members of my committee, Dr. John Harris, Dr. Jian Li, Dr. Jacob Hammer, and Dr. Howard Rothman, for their insightful comments which improved the quality of this dissertation.

Special thanks go out to all the CNEL students, some but a distant memory now, but especially Frank Candocia and John Fisher.

Finally, I also wish to thank Cindy Stark for her support, as well as my cats, Tiger and Trixie, even though they'll never read this, for providing welcome diversion during the long evening hours when most of this was written.

TABLE OF CONTENTS

	<u>Page</u>
ACKNOWLEDGEMENTS	II
ABSTRACT	VI
CHAPTERS	
1 INTRODUCTION	1
1.1 Goal	1
1.2 Motivation	2
1.3 Outline	2
1.4 Stationarity	3
1.5 Ergodicity	4
1.6 Piecewise Stationarity	6
1.7 Tracking vs. Dividing a Piecewise Stationary Signal	7
2 CLASSICAL METHODS OF SEGMENTATION	10
2.1 Classical Methods of Segmentation	10
2.2 Tests for a Change in First Order Statistics	11
2.3 Computational Complexity	14
2.4 Estimation of the Parameters	14
2.5 Tests for a Change in Higher Order Statistics	15
2.6 Simulations	17
3 A FRAMEWORK FOR UNSUPERVISED GATED COMPETITIVE SYSTEMS	21
3.1 Introduction	21
3.2 Gated Competition of Experts	22
3.3 Input vs. Output Based Gating	24
3.4 Annealed Competition of Experts	24
3.5 Mixture of Experts	27
3.5.1 Training with the EM Algorithm	28
3.5.2 Training with Gradient Descent	31
4 COMPETITIVE PRINCIPAL COMPONENT ANALYSIS	32
4.1 Introduction	32
4.2 Principal Component Analysis	32

	<u>Page</u>
4.3 Competitive PCA Using the Mixture of Experts	34
4.4 The PDF of Reconstruction Error	35
4.5 Solution of Competitive PCA Decomposition	43
4.6 Practical Implementation Issues	46
4.6.1 Applicability	46
4.6.2 Network design	46
4.6.3 Initialization	47
4.6.4 Training Issues	48
4.6.5 Repeatability and convergence	48
4.7 Extensions	48
4.7.1 Minimum eigenvalue approach	48
4.7.2 Modeling Power Variations Within a Regime	49
4.7.3 Principal Sub-space Decomposition	50
4.7.4 Reducing Spurious Gate Switching	50
4.7.5 On-line learning	50
4.8 Application to Images	51
4.8.1 Texture Segmentation	51
4.8.2 Real Image Segmentation	55
4.9 Conclusion	59
5 TEMPORAL PRINCIPAL COMPONENT ANALYSIS	60
5.1 Application of Competitive PCA to Time Series	60
5.2 Competitive PCA Simulations	61
5.2.1 Signal Segmentation: Switching FIR Process	61
5.2.2 Signal Segmentation: Earthquake Seismogram	62
5.2.3 Noise Reduction	63
5.3 Temporal PCA	77
5.3.1 The Large Window Solution	78
5.3.2 The Small Window Solution	78
5.4 Motivation: Matched Filter Approach	79
5.5 Separable Kernel Approach	82
5.6 Expansion in Terms of Multiple Frequency Shifted DPSWF	85
5.7 Conclusion	89
6 MEMORY IN GATED COMPETITIVE SYSTEMS	90
6.1 Introduction	90
6.2 Adding Memory to a Cost Function	90
6.3 Neighborhood Map of Competing Predictors	92
6.3.1 Simulation I: Switching FIR Process	93
6.3.2 Simulation II: A Non-stationary Time Series	96
6.3.3 Simulation III: Phoneme Segmentation of Speech	97
6.4 Self-annealing Competitive Prediction	98
6.4.1 Coupling the Competition with the Memory Depth	98
6.4.2 Gradient Descent	100
6.4.3 Simulation	101
6.5 Adding Static Memory to the Mixture of Experts	102

	<u>Page</u>
6.5.1 Adding Static Memory to the Experts' pdf	103
6.5.2 Chi-Square Distribution	103
6.5.3 Gradient Descent	103
6.5.4 Simulation	105
6.6 Adding Recursive Memory to the Mixture of Experts	107
6.6.1 Gamma Distribution of the Recursive Mean Square Error	107
6.6.2 Gradient Descent	111
7 CONCLUSIONS AND FUTURE WORK	113
7.1 Comparison of Present Work to Classical Segmentation	113
7.2 Comparison of Input and Output Based Competitive Systems	115
7.3 Competitive PCA	116
7.3.1 Application to Images	117
7.3.2 Application to Time Series	117
7.4 Temporal PCA and the Discrete Prolate Spheroids	117
APPENDIX: PROPERTIES OF THE DISCRETE PROLATE SPHEROIDS	119
REFERENCES	124
BIOGRAPHICAL SKETCH	129

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

GATED COMPETITIVE SYSTEMS FOR UNSUPERVISED SEGMENTATION
AND MODELING OF PIECEWISE STATIONARY SIGNALS

By

Craig Langdale Fancourt

December 1998

Chairman: Dr. José C. Principe

Major Department: Electrical and Computer Engineering

The major goal of this dissertation is to present a new paradigm for the unsupervised competitive segmentation and modeling of signals into stationary segments, and to elucidate and develop several specific algorithms that fall within the paradigm. The identification of the segments is accomplished by several experts which act in parallel on a local section of the signal at a time. The experts can utilize any unsupervised learning algorithm to which one can ascribe a performance measure, the two most common of which are the tasks of prediction and auto-association. The experts learn in the usual way except that the local data are weighted based on how well the experts have performed in the past. This information is provided by the gate, which analyzes the performance of the experts and then ascribes to each a relative measure of the validity of each expert to that part of the data. The cumulative effect of these gate assignments is to segment the data.

A gate can be either input or output based. An input based gate attempts to learn to predict the validity of the experts from the input only, thus eventually eliminating the need for performance information, while an output based gate always analyzes the performance of the experts.

We use principal component analysis (PCA) experts, which requires finding the probability density function for auto-association, and then show that the resulting learning equations closely parallel those of a single PCA expert.

When applied to time series, PCA becomes temporal PCA. We show that the discrete prolate spheroidal wave functions form a natural basis for the eigenfunctions in the frequency domain, and that this basis explains most of the properties of temporal PCA.

Finally, we show that momentum learning is equivalent to a recursive mean square error estimator, and then use this estimator to add memory to the gate, which improves segmentation and learning by giving greater validity to experts that have performed well in the recent past.

CHAPTER 1 INTRODUCTION

1.1 Goal

The goal of this dissertation is to develop a new class of algorithms for non-stationary time series and image analysis through the segmentation of data into stationary regions and modeling of the resulting segments, all in a completely unsupervised fashion. One must always make some assumptions about the data with which one is working. Here, we assume that the data are piecewise stationary, with switching between dynamical systems occurring rapidly, but where the switching rate itself occurs at a much slower rate. The dynamical systems which produced the data are unknown, as are the switching times.

We need to give more precise definitions of some of the terms used, but first it is instructive to consider some real data that can be regarded as at least piecewise quasi-stationary. At a small enough time scale (5-100 msec) speech can be roughly regarded as a sequence of quasi-stationary segments. These segments can be considered quasi-stationary because, although slowly evolving, they have been fairly successfully modeled, as part of a larger speech recognition system, by simple linear auto-regressive (AR) systems with constant parameters, driven either by a pulse train, in the case of voiced speech, or noise, in the case of unvoiced speech. Although the primary focus of this dissertation is time series, most of what will be presented is equally applicable to images. Some textures within an image can be regarded as stationary in the sense that, on a certain scale, the local intensity correlations between neighboring pixels are relatively constant over the entire

texture. An image is considered piecewise stationary if the transitions between textures occur at well defined boundaries.

1.2 Motivation

There are many applications of segmentation and identification of signals. In monitoring any process, such as the operation of a jet or car engine, or a machine tool, there may be several stationary regimes, some of which represent in-process states, and others which represent out-of-process states. The detection of a change in stationarity can be a possible indication of an out-of-process state, and identification of the new regime can help determine this, and thus whether additional steps need to be taken.

In the area of control, a plant may have several stationary regimes, each of which requires a specially tuned controller. Detection of a change in the plant and identification of the new regime can be used to switch to the appropriate controller.

In some cases, identification of the stationary regime is the goal itself. In the case of speech recognition, identification of the quasi-stationary segments can be passed on to a higher level processor for word or sub-word recognition.

1.3 Outline

In the remainder of this chapter, we will define some of the terms and concepts laid out in the goal. In chapter two we will examine the classical approaches to segmentation and modeling. In chapter three we will present a unified framework for unsupervised gated competitive systems, under which fall all the new algorithms presented in this dissertation, and show that some recent algorithms in the literature also fall under the same framework. In chapter four we will examine gated competitive principal component analysis and apply

it to image segmentation. In chapter five we will apply gated competitive principal component analysis to time series, and then attempt to understand the results in terms of special functions called the discrete prolate spheroidal wave functions. In chapter six we will look at various ways of adding memory to gated competitive systems, as well as novel gate structures. Finally, in chapter seven we will draw conclusions on the present work and suggest possible lines of inquiry for future work.

1.4 Stationarity

We now consider a more technical definition of a stationary signal. To do so, we must first review the concept of a random process. Imagine sampling some property of a dynamical system over a period of time, so that we have a series of measurements, $x(t_i)$. Now, imagine that somehow we can reset the dynamical system and repeat the measurements. If the results are different, then the process is random and not deterministic. A good example of this is the many ways a single speaker can say the same word. Because of changes in stress, co-articulation, duration, and physical condition, no speaker ever says the same word in exactly the same way. If we repeat the experiment an infinite number of times, we can find the first order probability density function (pdf) of the signal at each sampled time instant, $p(x(t_i)) \forall i$. Now, $x(t_i)$ can be regarded as a random variable. A *first order stationary process* is one whose first order pdf is the same for all given sampled times: $p(x(t_i)) = p(x(t_j)) \forall i, j$. Note that the term “first order stationarity” refers to the number of random variables only; if the pdf’s are the same, all higher order moments must also be the same: $E[x^a(t_i)] = E[x^a(t_j)], \forall i, j$.

First order stationarity does not, however, say anything about the relationships across time between the random variables. For that, we need the second order or joint pdf

between any two sampled time instants, $p(x(t_i), x(t_j))$, $\forall i, j$. A *second order stationary process* is one where the joint pdf between two sampled time instants depends only on the time difference between them. That is, the joint pdf remains the same for all constant time differences: $p(x(t_i), x(t_j)) = p(x(t_m), x(t_n))$ when $t_i - t_j = t_m - t_n$. Second order stationarity implies that all joint moments will only be a function of the time difference: $E[x^a(t_i)x^b(t_j)] = f(t_i - t_j, a, b)$, $\forall i, j$.

The strictest stationarity requires that all higher order pdf's be invariant to uniform time shifts of the random variables. However, estimating joint pdf's from data is subject to the so-called dimensionality explosion, requiring an exponentially increasing amount of data with each increase in pdf order, thus rendering impractical higher order stationarity tests. Even estimating second order pdf's is difficult. For this reason, a lesser requirement known as *wide sense stationarity* is often employed, for which it is only necessary that the first moments of the first and second order pdf's be independent of time shifts: $E[x(t_i)] = E[x(t_j)]$ and $E[x(t_i)x(t_j)] = f(t_i - t_j)$, $\forall i, j$. Note that the latter equation implies that the second moment of the first order pdf will also be independent of time: $E[x^2(t_i)] = E[x^2(t_j)]$, $\forall i, j$. Because Gaussian processes are completely characterized by their first and second moments, a wide sense stationary Gaussian process is also second order stationary.

1.5 Ergodicity

There is another, more fundamental problem with these definitions of stationarity. They often cannot be tested in practice because many physical dynamical systems are uncontrolled free-running systems which cannot be reset, and thus there exists only one realization of the random process. We then have to hope that the time average statistics of

the single realization are equivalent to the process average statistics, in which case the process is called *ergodic*. Note that an ergodic process is stationary but a stationary process is not necessarily ergodic.

If we have a single realization of a first order ergodic process sampled at discrete time intervals, we can then estimate both the first order pdf $p(x)$ and second order joint pdf, $p(x(n), x(n+m))$, with the same confidence as for multiple realizations, providing we sample the data for a sufficiently long time. Joint moments, due to the implied second order stationarity, are a function of the discrete lag only: $E[x^a(n)x^b(n+m)] = f(m, a, b)$. The most common joint moment is the autocorrelation: $r(m) = E[x(n)x(n+m)]$. Note that we are often only interested in the correlations, and thus estimate them directly from the data, bypassing the problem of estimating the pdf's.

To be fair, there are cases where we can repeat an experiment and the ergodic assumption is not required. Certainly a speaker can repeat a word many times. However, even in such cases there is the problem of time alignment and warping. For example, without proper time alignment, the fact that a word is composed of quasi-stationary segments might be completely obliterated in repeated experiments. Furthermore, the large number of trials required for statistical significance is often impractical.

Finally, a note about human perception of stationarity, again using speech as an example. It is worth noting that a non-stationary signal may not necessarily be interpreted as such by humans. As an example, a change in volume in the middle of an otherwise quasi-stationary segment of speech is readily identified as conveying the same information, but in fact creates a non-stationary signal. This change in amplitude has increased the variance of the pdf but has not changed the generic shape of the pdf itself.

1.6 Piecewise Stationarity

Having defined stationarity, it is now natural to define piecewise stationarity. Generally, there are two frameworks that describe piecewise stationary signals. Common to both are a generic state space representation of a dynamic model described by a parameter set, θ , and an internal state \mathbf{x} . The model is driven by an excitation source, $u(n)$. A change in either the parameter set or the excitation source may create a new stationary regime.

The first framework is called the external switching framework and is diagrammed in Figure 1-1 (a). At a well defined transition time, the switch connects to a different dynamic model, producing a new stationary regime. The second is called the parameter switching framework and is diagrammed in Figure 1-1 (b). At the transition time, one parameter set or excitation is simply replaced by another.

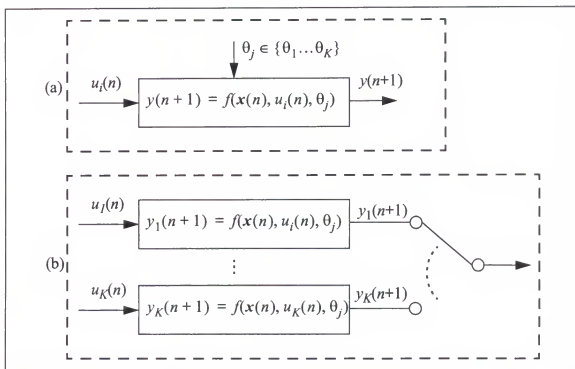


Figure 1-1. Piecewise stationary models: (a) parameter switching; (b) external switching.

The primary difference between these two models are the transient effects. If the dynamic model has memory, then the parameter switching model will exhibit transient effects for the effective memory depth of the system. The external switching model, on the other hand, will never exhibit transient effects. Clearly the external switching model is an idealization.

There is another class of signals that are called quasi-stationary signals. In these signals the transition between regimes occurs gradually rather than instantaneously. The external switching model can be altered to produce a quasi-stationary signal by replacing the switch with a gradually changing mixture during the transition period. The parameter switching model can be altered to allow the dynamic model parameters to gradually evolve over time from one set to another. In fact, an adiabatic transition between different parameter sets will generally exhibit less severe transient effects than a quantum switch.

Using speech as an example again, in a transition between two voiced speech segments the excitation remains nearly the same, but the shape of the vocal tract changes, resulting in a continual change in the parameter set of the corresponding dynamic model.

1.7 Tracking vs. Dividing a Piecewise Stationary Signal

The question now becomes, given our goal of segmenting and modeling piecewise stationary signals, how can we accomplish this? When confronted with a non-stationary signal that changes its 2nd order or higher statistics, the normal approach is to employ a single adaptive system. An adaptive system has continually adjusting parameters that respond to the local (in time) properties of the signal. In order to be local in time, an adaptive system must have a finite effective memory. That is, it must eventually forget what it has learned in the past. The effective memory of an adaptive system may be governed by a learning

rate, for example, in the case of the least means square (LMS) filter, or a “forgetting factor”, in the case of a recursive least squares (RLS) filter. However, this finite memory creates a classic trade-off between the speed of convergence and the misadjustment after convergence. Misadjustment describes the variance of the parameters about their nominal solution. In terms of the segmentation and modeling problem, this translates into a trade-off between the accuracy of the modeling and the delay in detecting a new stationary regime. If the effective memory is too short, the parameters exhibit a large variance about the mean, which can even masquerade as a transition. If the effective memory is too long, transition times become uncertain or worse, short stationary regions may be missed entirely.

There is, however, a more fundamental limitation in using a single adaptive system to monitor a non-stationary signal. The normal procedure for detecting a change involves monitoring the innovations of the adaptive system for deviation from whiteness, or a change in variance. The innovations are the part of the signal that cannot be explained by the model. Unfortunately, it can be shown that the sequence of innovations of a single system is an *insufficient statistic* (see Basseville and Nikiforov, 1993). That is, there is a many to one mapping from different stationary regimes to identical statistics of the innovations. For this reason, two or more adaptive models are required, and this forms the basis of the classical methods of segmentation to be presented in chapter two.

The classical methods are sequential in nature, in that information from prior stationary regions is discarded. Under the production models of Figure 1-1, such methods are appropriate whenever the potential number of dynamical models is very large, so that there is a small probability of returning to the same model. In this case, it is appropriate to

view the data as a sequence of stationary *regions*, and the segmentation problem becomes one of detecting the *transition* between two stationary regions.

There is another, fundamentally different, approach to dealing with piecewise stationarity. Instead of viewing the data as a sequence of stationary *regions*, we model the data under the assumption that it has been produced by switching among a finite set of stationary *regimes*. This approach is only feasible when the number of potential regimes is reasonably small.

Modeling the regime instead of the region has a number of advantages. First and foremost, if several discontinuous regions belong to the same regime, information from all these regions can be used to improve the model estimate. Second, a segmentation system that models the regimes can generalize to new piecewise stationary processes as long as they are generated by the same switching model.

The development of various algorithms that segment a signal by modeling the regimes forms the fundamental topic of this dissertation.

CHAPTER 2 CLASSICAL METHODS OF SEGMENTATION

2.1 Classical Methods of Segmentation

Consider the following problem: given a single realization of a piecewise ergodic random process, consisting of a finite ordered set of vectors, segment the data into contiguous stationary regions. This is known as the off-line segmentation problem. We do not know how many stationary regions are contained in the data, nor the change points between regions. Because of this, the problem cannot be easily formulated as a global test between known hypotheses. The only possible approach then is to perform sequential *detection* of change points, which makes the off-line segmentation problem similar to the on-line case, reducing the problem to one of detecting successive change points between successive pairs of stationary regions. This problem was first addressed by Page (1957) but the algorithm we will now examine in some detail was first presented by Brandt (1982, 1983).

In the absence of any prior information, the classical approach in this case is a generalized likelihood ratio (GLR) test for deciding between two hypotheses: the null hypothesis, H_0 , that no change occurred within N samples, and the posited hypothesis, H_1 , that a change did occur at the intermediate sample time T . From the likelihood ratio of these two hypotheses, a decision function is found that forms a change detector. When the decision function exceeds a preset threshold, a change is detected. The exact transition time is then determined, and the algorithm is reset and started anew.

2.2 Tests for a Change in First Order Statistics

Consider the problem of detecting a change in the first order pdf of an independently identically distributed (i.i.d.) process, assuming knowledge of the parametric form of the pdf, $p_{\theta}(x)$, characterized by the parameter set θ . Since we are interested in an unsupervised algorithm, we do not know the parameters of the pdf before the change point, θ_0 , or after the change point, θ_1 , nor the change time, T . The likelihood of the null hypothesis, H_0 , that no change occurred within N samples is

$$L(H_0) = \prod_{n=1}^N p_{\theta_0}(x(n)). \quad (2.1)$$

Likewise, the likelihood of the hypothesis H_1 that a change in the pdf occurred at time T is given by

$$L(H_1) = \left[\prod_{n=1}^{T-1} p_{\theta'_0}(x(n)) \right] \cdot \left[\prod_{n=T}^N p_{\theta_1}(x(n)) \right]. \quad (2.2)$$

Note that we differentiate between the parameter set under the null hypothesis, θ_0 , and the parameter set before change under the change hypothesis, θ'_0 , since they are estimated over different times. The log-likelihood ratio between the two hypotheses is then

$$S_1^N = \log \left[\frac{L(H_1)}{L(H_0)} \right] = \log \left[\frac{\prod_{n=1}^{T-1} p_{\theta'_0}(x(n)) \prod_{n=T}^N p_{\theta_1}(x(n))}{\prod_{n=1}^N p_{\theta_0}(x(n))} \right] \quad (2.3)$$

which can also be written

$$S_1^N = \sum_{n=1}^{T-1} \log \left[\frac{p_{\theta'_0}(x(n))}{p_{\theta_0}(x(n))} \right] + \sum_{n=T}^N \log \left[\frac{p_{\theta_1}(x(n))}{p_{\theta_0}(x(n))} \right]. \quad (2.4)$$

The decision function is then formed by replacing the parameters θ_0 , θ'_0 , and θ_1 , in S_1^N by their maximum likelihood (ML) estimates, $\hat{\theta}_0$, $\hat{\theta}'_0$, and $\hat{\theta}_1$, over their respective regions and then maximizing that with respect to the change time T

$$g(N) = \max_T [S_1^N |_{\hat{\theta}_0, \hat{\theta}'_0, \hat{\theta}_1}] = \max_T \max_{\theta_0} \max_{\theta'_0} \min_{\theta_1} [S_1^N]. \quad (2.5)$$

A change is deemed to have occurred within the block of N data if the decision function exceeds some predefined threshold

$$g(N) \underset{H_0}{\overset{H_1}{\geq}} \lambda. \quad (2.6)$$

From (2.4) and (2.5), we see that under H_0 , the decision function tends to fluctuate near zero, and under H_1 , the decision function increases with increasing N .

To summarize, the algorithm starts from the last detected change point, from which a small block of N points is examined. To find $g(N)$, the two hypotheses are compared using (2.4) for all possible changepoints, T , within the block, and the maximum value of S_1^N is assigned to $g(N)$. If $g(N)$ is less than the threshold, no changepoint is detected, N is incremented, and the process repeated. If $g(N)$ is greater than the threshold, the algorithm is reset and started anew. A symbolic diagram of this is shown in Figure 2-1 (a). Regions where the number of samples is effected by T are shown with solid arrows, while regions effected by N are shown with hollow arrows.

As an example, consider the problem of detecting a change in the mean, μ , of an independent identically distributed (i.i.d.) Gaussian process under the assumption of constant

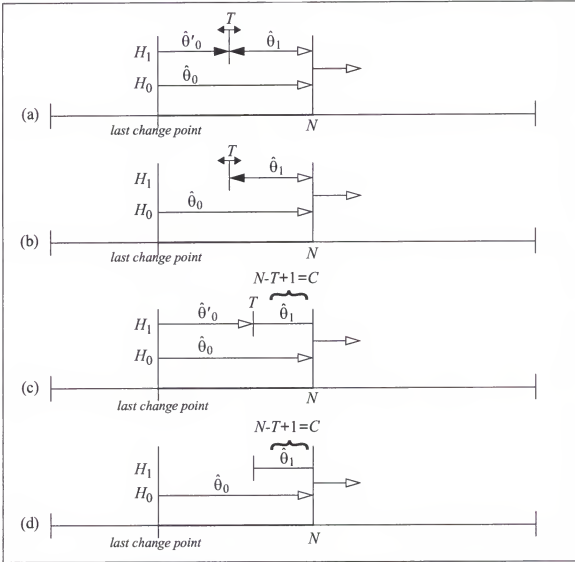


Figure 2-1. Classical change point detection: (a) three model; (b) two model; (c) three model with fixed length region for θ_1 ; (d) two model with fixed length region for θ_1 .

variance. The maximization with respect to the parameters in (2.5) is explicit, resulting in the following decision function

$$g(N) = \max_T [(T-1)(\mu'_0)^2 + (N-T+1)(\mu_1)^2 - N(\mu_0)^2]. \quad (2.7)$$

Using the relationship between the means, $N\mu_0 = (T-1)\mu'_0 + (N-T+1)\mu_1$, this can be further simplified to

$$g(N) = \max_T \left[\frac{N(N-T+1)}{T-1} (\mu_1 - \mu_0)^2 \right]. \quad (2.8)$$

Note that the relationship between the model parameters allowed reformulation of the three model decision function in terms of two models.

2.3 Computational Complexity

This algorithm is computationally intensive. For each block of N points, on the order of $N+2$ new ML estimates must be performed. Extensive use of recursive ML estimates can significantly reduce the computational load. However, (2.4) must still be evaluated N times. A minor simplification can be made by assuming that the null maximum likelihood estimates are approximately equal, $\hat{\theta}'_0 \approx \hat{\theta}_0$, resulting in the two model GLR algorithm shown in Figure 2-1 (b). The first term in (2.4) is then zero, and the whole equation simplifies to

$$S_1^N = \sum_{n=T}^N \log \left[\frac{p_{\hat{\theta}_1}(x(n))}{p_{\hat{\theta}_0}(x(n))} \right]. \quad (2.9)$$

Note that $\hat{\theta}_0$ is still estimated over all N samples but the GLR is only evaluated over the last $N-T+1$ samples.

A more dramatic reduction can be achieved by eliminating the maximization over T in (2.5) by fixing the length, $N-T+1$, of the data segment associated with the parameter set θ_1 . This results in the symbolic diagram shown in Figure 2-1 (c). Finally, these two simplifications can be combined, resulting in the symbolic diagram of Figure 2-1 (d). Eliminating the maximization over T also allows for on-line recursive estimation of the parameters.

2.4 Estimation of the Parameters

Having *detected* a change, the problem is then transformed into one of *estimation* of the parameter sets and change point. Given that a change was detected in a sub-block of

data of size N , the problem is to find the ML estimates of the unknown parameters sets and the change time

$$(\hat{T}, \hat{\theta}_0, \hat{\theta}_1) = \max_{T, \theta_0, \theta_1} \left[\log \left[\prod_{n=1}^{T-1} p_{\theta_0}(x(n)) \prod_{n=T}^N p_{\theta_1}(x(n)) \right] \right]. \quad (2.10)$$

Again replacing the parameter sets by their maximum likelihood estimates over their respective regions, this can be simplified to

$$\hat{T} = \arg \max_T \left[\sum_{n=1}^N \log[p_{\hat{\theta}_0}(x(n))] + \sum_{n=T}^N \log \left[\frac{p_{\hat{\theta}_1}(x(n))}{p_{\hat{\theta}_0}(x(n))} \right] \right]. \quad (2.11)$$

Note that the first term does not depend on T , and thus can be ignored. What remains is identical to the decision function, $g(N)$, for the two model case. This means that the estimation of the parameters can be done in conjunction with the detection.

The close relationship between detection and estimation also applies to the on-line and off-line implementations. In the on-line case the algorithm is usually re-started from the point of *detection*, which inevitably involves a delay from the changepoint. In the off-line case, the algorithm is usually re-started from the ML *estimate* of the changepoint.

2.5 Tests for a Change in Higher Order Statistics

The previous formulation does not work when trying to find change points in a higher order joint pdf because the process is no longer i.i.d. However, if we can find an inverse model for each of the stationary regimes, the output of the inverse filter, the so called innovations, is i.i.d. Then, all of the previous development still holds and the problem becomes one of detecting a change in the pdf of the innovations. The off-line approach to detecting a change in an ARMA process was first investigated by Kligiene and Telksnys (1983).

When the model that produced the signal is known, the inverse model can be found analytically. However, in most cases, we do not know the original model. In this case, the inverse model can be found from the data and an adaptable model that learns to predict the next value, as shown in Figure 2-2. If the parameter set is adjusted so that the innovations are completely unpredictable, then all possible information in the data has been extracted. Then, the overall transfer function from input to residuals must be the inverse of the transfer function of the model that produced the data. Note that if the excitation of the model that produced the data has a predictable component, that will be modeled as well by the predictor.

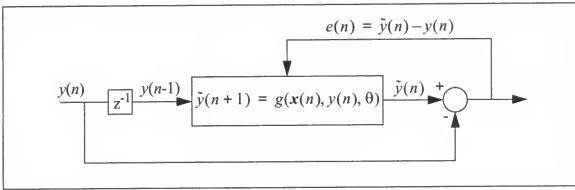


Figure 2-2. Using prediction to convert a higher order process to an i.i.d. process.

We now consider the problem of detecting a change in the residuals when their pdf is Gaussian. It is easy to show that the decision function in the three model case is

$$\begin{aligned}
 g(N) &= \max_T [N \log[\hat{\sigma}_0^2] - (T-1) \log[\hat{\sigma}_0^2] - (N-T+1) \log[\hat{\sigma}_1^2]] \\
 &= \max_T \left[(T-1) \log \left[\frac{\hat{\sigma}_0^2}{\hat{\sigma}_1^2} \right] + (N-T+1) \log \left[\frac{\hat{\sigma}_0^2}{\hat{\sigma}_1^2} \right] \right] \quad (2.12)
 \end{aligned}$$

where $\hat{\sigma}_0^2$, $\hat{\sigma}_0^2$, and $\hat{\sigma}_1^2$, are the variances of the residuals of the models over their respective regions. Note that, unlike the first order case where there is usually a simple relation-

ship between the parameters, there is no simple way to relate the model variances to each other. As in the first order pdf test, the decision function tends to fluctuate slightly above zero until a change is detected, at which point it rapidly increases. Also, the same simplifications can be made to reduce the computational load.

The decision function in (2.12) is exactly the same formulation for detecting a change in the variance of a first order Gaussian pdf. Thus, this test cannot distinguish between a change in the signal power and a change in the model. Rather than use (2.12) and perform additional tests to determine the type of change, some authors have suggested decision functions that directly incorporate other distance measures between models. In the case of ARMA modeling, the L2 norm between the spectrums can be used. For the more general case, various entropy measures between the pdf's can be employed.

2.6 Simulations

We now present results of the three model GLR segmentation algorithm for many of the time series to be presented later in this dissertation. For each figure on the following pages, the top panel shows the original time series along with vertical lines representing the known changepoints, as determined either by design or human expert scoring. The bottom panel shows the decision function, with vertical lines representing the detected changepoints. Each figure caption includes information on the order (of the AR predictors), threshold (for detection), and deadzone (the number of samples added after a detection). In addition, some of the longer time series have an increment (the number of samples added after no detection). After a detection, the algorithm was reset from the point of detection.

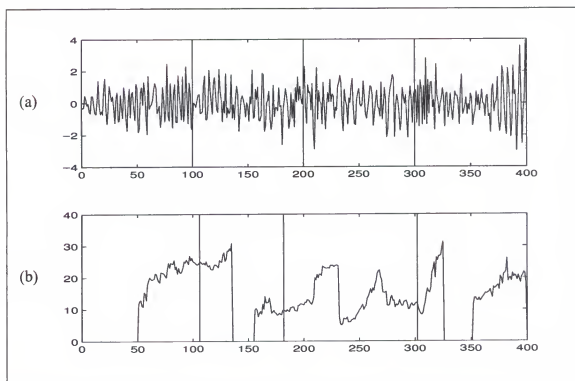


Figure 2-3. Switching FIR process: (a) time series and segmentation; (b) decision function and segmentation (order = 8, threshold = 30, deadzone = 50).

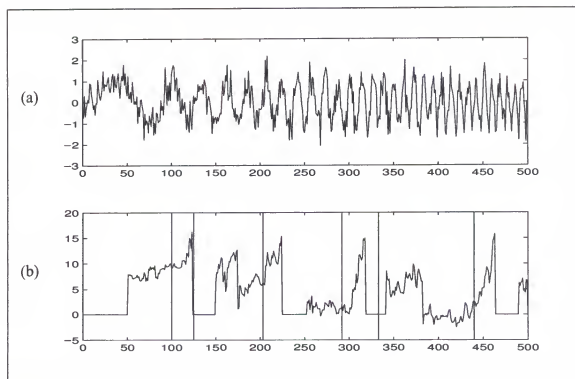


Figure 2-4. Noisy chirp signal: (a) time series; (b) decision function and segmentation (order = 8, threshold = 15, deadzone = 50).

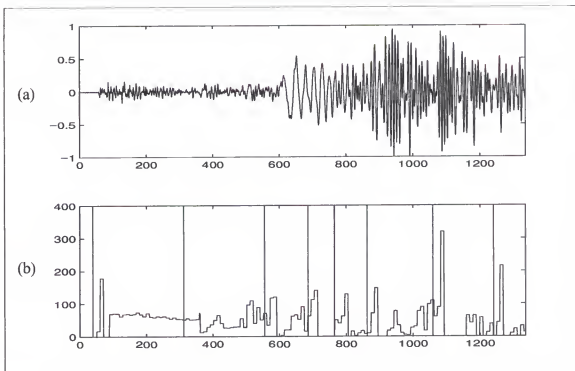


Figure 2-5. Earthquake seismogram: (a) time series; (b) decision function and segmentation (order = 8, threshold = 120, deadzone = 20, increment = 10).

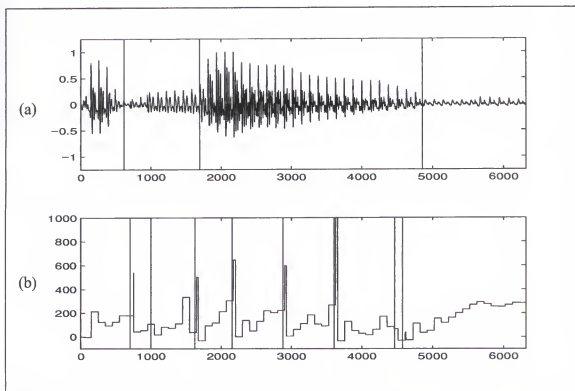


Figure 2-6. Speech signal: (a) time series and segmentation; (b) decision function and segmentation (order = 12, threshold = 400, deadzone = 50, increment = 100).

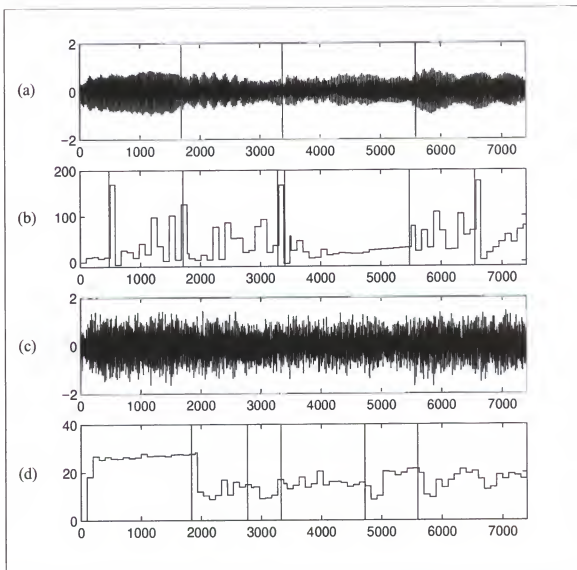


Figure 2-7. Violin: (a) time series and segmentation; (b) decision function and segmentation (order = 8, threshold = 100, deadzone = 100, increment = 100); (c) noisy time series; (d) decision function and segmentation (order = 8, threshold = 30, deadzone = 100, increment = 100).

CHAPTER 3

A FRAMEWORK FOR UNSUPERVISED GATED COMPETITIVE SYSTEMS

3.1 Introduction

We saw in the previous chapter how classical likelihood ratio techniques can be used to segment and model a piecewise stationary signal in an unsupervised fashion. In the absence of apriori information about the number of stationary regimes, the classical techniques are necessarily sequential in nature. That is, after the last detected change point, they concentrate entirely on the current stationary region, continually testing for a transition to a potentially new stationary region, and do not use any information from prior regions.

We now consider the case where the stationary regions are due to switching between a *finite set* of unknown generative models. We still consider the piecewise stationary case, where only one model is active at a given time, but now a single model can be repeatedly active at different periods in time. Therefore, in designing algorithms for this case, instead of concentrating on stationary *regions*, we need to concentrate on the stationary *regimes* that produced the data.

To this end, we now introduce the competitive multiple model approach, where several adaptable “experts” compete to explain the same data. The most successful experts are granted larger parameter updates. As an expert begins to specialize on a particular stationary region, it is then at a disadvantage for other stationary regions. Such an approach has two key components: an evaluation *mechanism*, through which the relative performance of

the experts can be measured, and a competitive *framework*, which governs the competition between experts by comparing the experts' performance and relating that to their adaptation to the data. The competitive framework is implemented as a gate, and the whole architecture called "gated experts". It is the role of the gate to determine which expert is valid at the present time. The history of the gate's decisions effectively segments the data. It is the role of the experts to learn when the gate instructs them to do so, and when they are not learning, to remember the modeling of regimes they have learned in the past.

3.2 Gated Competition of Experts

In the activation phase, the total system output is a weighted sum of the experts' outputs

$$y = \sum_{k=1}^K g_k y_k(x) \quad (3.1)$$

where y_k is the output of the k^{th} expert with x as input, and g_k is the k^{th} output of the gate. In order for the mixture to be meaningful, the gate output is usually constrained to sum to one

$$\sum_{k=1}^K g_k = 1. \quad (3.2)$$

Such a linear mixture of experts can represent either a competitive or cooperative system, depending on how the system learns, as specified by the cost function. In the context of introducing their Mixture of Experts model, Jacobs et al. (1991) first presented a cost function that encourages competition among gated expert networks, which we generalize to

$$c = \sum_{k=1}^K g_k f(d - y_k(x)) \quad (3.3)$$

where d is the desired signal. The *evaluation mechanism* is embodied in the term $f(d - y_k)$, a function of the performance or error of the k^{th} expert. Since the desired signal is the same for all experts, they all try to regress the same data, and are always in competition. This alone, however, is not enough to foster specialization. The *competitive framework* is embodied in the gate g_k , which represents some measure of the k^{th} expert being the correct one.

Typical functions for $f()$ include the squared error, in which case the total cost function over a data set of N samples is the sum of the instantaneous costs

$$C = \sum_{n=1}^N \sum_{k=1}^K g_k(n) [d(n) - y_k(x(n))]^2 \quad (3.4)$$

or a probability density function in the error which, if i.i.d., results in the total cost function as a product of the instantaneous costs

$$C = \prod_{n=1}^N \sum_{k=1}^K g_k(n) p(d(n) - y_k(x(n))). \quad (3.5)$$

The formalism represented by (3.3) is a supervised algorithm, in that it requires a desired signal. However, we are interested in a completely unsupervised algorithm for performing segmentation and modeling. A supervised algorithm becomes unsupervised when the desired signal is a fixed transformation of the input. Although many transformations are possible, the two most common transformations involve the delay operator and the identity matrix, resulting in *prediction* and *autoassociation*, respectively. Although not

always explicitly stated, throughout this work it is to be understood that the indication of a desired signal implies some transformation of the input $d \Rightarrow d(x)$.

3.3 Input vs. Output Based Gating

All the algorithms embodied in (3.3) fall into two broad categories, which we designate as input or output based gating, and are shown in Figure 3-1. For output based gating, the gate is a direct *calculated* function of the performance, and hence, the *outputs*, of the experts. The simplest scheme for the gate that falls within this category is hard competition, for which the gate chooses the expert with the smallest magnitude error in a winner take all fashion. Other output based gates possess memory to keep track of past expert performance. Examples of algorithms that fall into this category include the Annealed Competition of Experts (ACE) of Muller et al. (1995), Xu's (1994) algorithm, and Self-Annealing Competitive Prediction of Fancourt and Principe (1997a), which we will present in chapter six. A more elaborate gate with both memory and a spatial structure is used in the Neighborhood Map of Competing Predictors of Fancourt and Principe (1996a), also to be presented in chapter six.

With input based gating, the gate is an *adaptable* function of the *input* that learns to forecast which expert will perform the best. The best example of input based gating is the Mixture of Experts (MOE) algorithm.

3.4 Annealed Competition of Experts

Perhaps the best known example of output based gating is the Annealed Competition of Experts (ACE) algorithm of Muller et al. (1995). A set of K expert neural predictors operate in parallel, and a gating function determines the probabilities of the experts. The

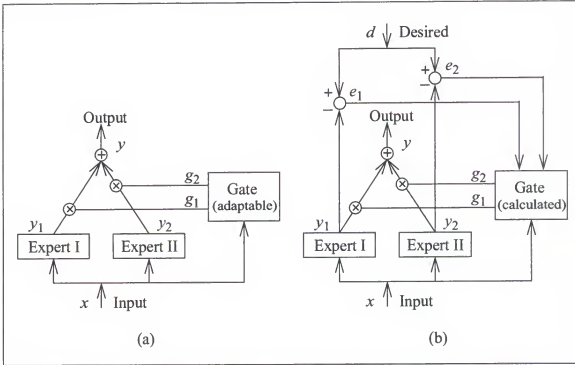


Figure 3-1. Two classes of gates during activation: (a) input based; (b) output based.

total cost function is based on (3.4) which uses the standard squared error as the evaluation mechanism. The competitive framework provided by the gate is a Gaussian function of the local mean square error

$$g_k(n) = A e^{-\beta \varepsilon_k(n)} \quad (3.6)$$

where A is a normalization constant, β is an annealing term which determines the degree of competition, and ε_k is related to the local mean square error of the k^{th} expert, calculated using a non-causal boxcar filter

$$\varepsilon_k(n) = \sum_{\tau=-\Delta}^{\Delta} e_k^2(n-\tau) = \sum_{\tau=-\Delta}^{\Delta} [d(n-\tau) - y_k(n-\tau)]^2. \quad (3.7)$$

The gating function is *calculated but not adapted*. Furthermore, the gradient of the cost function with respect to some weight, w_k , in the k^{th} expert, is calculated treating the gate as a constant

$$\Delta w_k = -\eta \frac{\partial C}{\partial w_k} = -\eta \sum_{n=1}^N g_k(n) e_k(n) \frac{\partial y_k}{\partial w_k} \quad (3.8)$$

where η is the learning rate. Thus, the gate effectively represents a weighting term to the learning rate of the individual experts.

The algorithm must be annealed during training. By starting the learning process with a small value of β , all the predictors see the entire data set and converge to an average of all the dynamical systems. Then as β is increased and the competition becomes harder, each predictor begins to specialize on a single stationary region of the input, which then puts it at a disadvantage for other stationary dynamical regions, allowing other predictors to win those regions. Muller et al. report “phase transitions”, specific values of β for which a dramatic change in the cost occurs as the experts begin to specialize. There may be several such phase transitions in the course of training. As $\beta \rightarrow \infty$, the competition becomes hard, and only the winning predictor earns the right to update its prediction, at which point the experts are “fine tuning” their modeling of their respective regimes.

Using non-linear neural predictors, they have successfully applied this algorithm to the segmentation and identification of switching chaotic maps and speech. However, the algorithm is still dependent on manual fine tuning to determine experimentally the memory depth of the moving average filter, and the annealing rate.

There are two key issues that are essential for the successful application of the algorithm. The first is the memory term, in the form of a lowpass “boxcar” filter, for the instantaneous squared error (3.7). In cases where there is a large overlap between the return maps of the various dynamical regimes, or a small signal to noise ratio, a single time instance is insufficient to estimate the probabilities of the experts. This can result in rapid

switching between predictors. However, under the assumption of a relatively slow switching rate, a single predictor would be expected to win for a longer time period. Increasing the memory depth gives a greater advantage to the predictor that has won in the recent past. The acausal nature of the filter is not an hindrance because of the off-line nature of the algorithm, but ensures that regime switches indicated by the gating function are aligned with the data.

3.5 Mixture of Experts

The Mixture of Experts (MOE) algorithm is an example of input based gating. We will now cover it in some detail since it has become one of the most popular competitive algorithms, but also because we will use it as the framework for competitive principal component analysis in the next chapter. The particular version we will examine is called the non-linear gated experts, after Weigend (1995a), for which the gate is a multi-layer perceptron. For the MOE, the cost function (3.3) is viewed as a statistical mixture model

$$p(d|x) = \sum_{k=1}^K P(k|x)p(d|x, k) = \sum_{k=1}^K g_k(x)p(d|x, k) \quad (3.9)$$

where the k^{th} gate output, $g_k(x)$, represents the apriori probability of the k^{th} expert and $p(d|x, k)$ is the conditional pdf of the desired signal, given the input and the parameters of the k^{th} expert. For regression, $p(d|x, k)$ is usually modeled as a Gaussian distribution in the error between the desired signal and the k^{th} expert's output

$$p(d|x, k) = \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp \left[-\frac{1}{2} (d - y_k)^T \Sigma_k^{-1} (d - y_k) \right] \quad (3.10)$$

where Σ_k is a free covariance parameter to be determined.

3.5.1 Training with the EM Algorithm

We now show how to train the MOE architecture using the Expectation-Maximization (EM) algorithm in an off-line fashion. In the following development, for ease of reading, we leave out explicit functional dependence. Thus, it is assumed that $g_k(n) \Rightarrow g_k(x(n))$ and $y_k(n) \Rightarrow y_k(x(n))$. Also, there is an implicit iteration index in all the following equations. Given a finite data set consisting of N input vectors and N desired vectors, we are now in a position to derive equations for training the system. The free parameters to be determined are the weights of the gating and expert networks, and the experts' covariances. Following Weigend et al. (1995a), the total likelihood over the entire data set is given by

$$L = \prod_{n=1}^N \sum_{k=1}^K g_k(n) p(d(n)|x(n), k) = \prod_{n=1}^N \prod_{k=1}^K [g_k(x(n)) p(d(n)|x(n), k)]^{I_k(n)} \quad (3.11)$$

where, under the assumption that only one expert is valid at a given time, the binary indicator variable, I_k , indicating which expert is valid, allowed us to replace the sum of the likelihoods over the experts by a product. The indicator variable is “missing”, in the sense that we do not know apriori which expert is valid at any time step. In the E step of the EM algorithm, for a given set of free parameters of the experts and gate, the entire data set is evaluated, holding the free parameters constant, to determine $\tilde{x}(n)$, $p(d(n)|x(n), k)$ and $g_k(n)$, for all k and n . We then replace the indicator variables, I_k , at every time step, by their expected value

$$h_k \equiv E[I_k] = P(k|x, d) = \frac{P(k|x)p(d|x, k)}{\sum_{j=1}^K P(k|x)p(d|x, k)} = \frac{g_k p(d|x, k)}{\sum_{j=1}^K g_j p(d|x, k)} \quad (3.12)$$

Thus, h_k is the posterior probability of expert k , given both the input and output.

For the M step, (3.11) is maximized or equivalently, the negative log-likelihood

$$C = \sum_{n=1}^N \sum_{k=1}^K \left\{ -h_k(n) \log[g_k(n)] + \frac{1}{2} h_k(n) [(d - y_k)^T \Sigma_k^{-1} (d - y_k) + \log|\Sigma_k|] \right\} \quad (3.13)$$

is globally minimized over the free parameters. The process is then repeated. If, in the M step, (3.13) is only decreased and not minimized, then the process is called the Generalized EM (GEM) algorithm. This is necessary when either the experts or gate is non-linear, and a search for the global minimum is impractical.

The first term in the summation of (3.13) can be regarded as the cross entropy between the posterior probabilities and the gate. It has a minimum when only one expert is valid, and thus encourages the experts to divide up the input space. In order to ensure that the outputs of the gate sum to unity, the output layer has a “softmax” transfer function

$$g_k = \frac{\exp[s_k]}{\sum_{j=1}^K \exp[s_j]} \quad (3.14)$$

where s_k is the k^{th} input to the softmax. For a gate implemented as a multilayer perceptron, the cross entropy term in (3.13) cannot be minimized in a single step and the GEM algorithm must be employed. If the gate is trained through gradient descent (backpropagation), the error backpropagated to the *input* side of the softmax, from (3.13) and (3.14), is

$$\frac{\partial C}{\partial s_k} = \sum_{n=1}^N g_k(n) - h_k(n). \quad (3.15)$$

This is the same equation that would result from a mean square error criteria if h_k is interpreted as the desired signal for the output, g_k , of a trainable network. Thus, the posterior probabilities act as targets for the gate.

As an aside, in Xu's (1995) algorithm, the gate for the next iteration is the average of the posterior probabilities for the prior iteration

$$g_k = \frac{1}{N} \sum_{n=1}^N h_k(n). \quad (3.16)$$

Thus, the gating is not adaptable but is calculated and thus is *output* based.

There is an analytical solution for the experts at the end of each iteration when they are linear regressors, $y_k = W_k x$, namely:

$$W_k = P_k R_k^{-1} \quad (3.17)$$

$$R_k = \frac{\sum_{n=1}^N h_k(n) x_k(n) x_k^T(n)}{\sum_{n=1}^N h_k(n)} \quad P_k = \frac{\sum_{n=1}^N h_k(n) d_k(n) x_k^T(n)}{\sum_{n=1}^N h_k(n)}. \quad (3.18)$$

Likewise, there is an exact solution for the covariance parameter Σ_k at the end of each iteration

$$\Sigma_k = \frac{\sum_{n=1}^N h_k(n) [d(n) - y_k(n)] [d(n) - y_k(n)]^T}{\sum_{n=1}^N h_k(n)}. \quad (3.19)$$

3.5.2 Training with Gradient Descent

When the experts are non-linear, or on-line learning is desired, the experts can be trained through gradient descent on some weight, w_k , in the k^{th} expert

$$\frac{\partial C}{\partial w_k} = - \sum_{n=1}^N \frac{g_k \frac{\partial}{\partial w_k} p(\mathbf{d}|\mathbf{x}, k)}{\sum_{j=1}^N g_j p(\mathbf{d}|\mathbf{x}, k)} = \sum_{n=1}^N \frac{h_k(n)}{\sigma_k^2} \left[e_k(n) \frac{\partial}{\partial w_k} y_k(n) \right]. \quad (3.20)$$

All other equations are unchanged.

CHAPTER 4 COMPETITIVE PRINCIPAL COMPONENT ANALYSIS

4.1 Introduction

As mentioned in the first chapter, any supervised algorithm can be converted to an unsupervised algorithm by making the desired signal a fixed transformation of the input. In this manner, Weigend (1995a) has shown that the normally supervised Mixture of Experts algorithm is capable of performing unsupervised segmentation and modeling of a time series when the experts are predictors of the original input one time step into the future. In this chapter, we seek an analogous development when the experts are linear auto-associators that reconstruct the input from a reduced dimensional space. In other words, we seek to combine principal component analysis (PCA) with the Mixture of Experts formalism. We first give a brief introduction to the terminology and notation and main results of PCA, without proof.

4.2 Principal Component Analysis

Consider expanding the random vector $x(n)$, of dimension D , in terms of some fixed but complete basis in the D dimensional space

$$x(n) = \sum_{i=0}^{D-1} z_i(n) u_i = U z(n) \quad (4.1)$$

where the matrix $U = [u_0 \ u_1 \ \dots \ u_{D-1}]$ is deterministic and full rank. If the basis is constrained to be orthonormal

$$\mathbf{u}_i^T \mathbf{u}_j = \delta_{ij} \quad (4.2)$$

multiplying both sides of (4.1) from the left by \mathbf{U}^T immediately yields the coefficients of the expansion

$$z_i(n) = \mathbf{u}_i^T \mathbf{x}(n) \Rightarrow \mathbf{z}(n) = \mathbf{U}^T \mathbf{x}(n). \quad (4.3)$$

If the expansion is performed over a subset of the basis functions, the result is an estimate of \mathbf{x}

$$\tilde{\mathbf{x}}(n) = \sum_{i=0}^{P-1} z_i(n) \mathbf{u}_i = \mathbf{U}_p \mathbf{z}_p(n) \quad P < D \quad (4.4)$$

where \mathbf{U}_p is a matrix formed from a subset of the columns of \mathbf{U} . Likewise, the subset of the expansion coefficients are given by

$$\mathbf{z}_p(n) = \mathbf{U}_p^T \mathbf{x}(n). \quad (4.5)$$

Given a set of random vectors, $\mathbf{x}(n)$, $n=1 \dots N$, what is the basis that minimizes the mean square reconstruction error over the data set, $E[\|\mathbf{x} - \tilde{\mathbf{x}}\|^2]$, under the same orthonormality constraints as in (4.2)? It is not difficult to show that the basis must satisfy the eigenvalue equation

$$\mathbf{R} \mathbf{u}_i = \lambda_i \mathbf{u}_i \quad (4.6)$$

where λ_i and \mathbf{u}_i are the eigenvalues and eigenvectors, respectively, of the autocorrelation matrix $\mathbf{R} = E[\mathbf{x} \mathbf{x}^T]$. The P eigenvectors are chosen on the basis of the P largest corresponding eigenvalues. The z_i are called the principal components. The minimum mean square reconstruction error is given by the sum of the discarded eigenvalues

$$\text{Min}[E[\|\mathbf{x} - \tilde{\mathbf{x}}\|^2]] = \sum_{i=P}^{D-1} \lambda_i. \quad (4.7)$$

4.3 Competitive PCA Using the Mixture of Experts

In our model, the experts are linear PCA networks. These experts compete for the same data on the basis of their performance. A natural measure of PCA performance is reconstruction error, which becomes the competitive mechanism. By analogy with auto-association, we propose to utilize the input itself as the desired signal, and the reconstructed version of the input as the output of each expert, so that (3.3) becomes

$$c = \sum_{k=1}^K g_k f(\mathbf{x} - \tilde{\mathbf{x}}_k) \quad (4.8)$$

where $\tilde{\mathbf{x}}_k$ is the reconstructed input by the k^{th} PCA expert. As previously discussed, there are several frameworks applicable to competitive PCA that can be described by (4.8). Dony and Haykin (1995) used hard competition with multiple PCA experts for image compression and segmentation. However, hard competition is extremely sensitive to initial conditions. We will utilize the non-linear gated experts, where the gate is a non-linear function of the input and $f(\cdot)$ is an appropriate pdf.

We now bring together the Mixture of Experts architecture and PCA. The basic architecture is shown in Figure 4-1, where each expert is a linear PCA network. Each expert has two outputs: the principal components, \mathbf{z}_k , which we call the *expert system output*, and the reconstructed input, $\tilde{\mathbf{x}}_k$, which we call the *expert training output*. It is rather important to understand why this choice was made. In PCA the goal is to either utilize the principal components or the eigenvectors themselves. But it is not the principal components that

drive the competition; rather, it is the reconstructions of the input by the experts. This is the reason the block diagram of Figure 4-1 differs from the more conventional MOE networks which display a single output. The gating outputs, g_k , weigh *both* the system outputs, z_k , and the training outputs, \tilde{x}_k . The total system and training outputs are thus given by

$$z = \sum_{k=1}^K g_k(x) z_k(x) \quad \tilde{x} = \sum_{k=1}^K g_k(x) \tilde{x}_k(x). \quad (4.9)$$

The gate is a multilayer perceptron which receives its inputs from the same delay line as the PCA experts.

Our first thought was to use the full set of PCA components to effect the reconstruction. However, each PCA expert is capable of perfectly reconstructing the input (the PCA transformation is full rank), so there is no information to drive the competition. In order to create a difference in performance between the PCA experts, to drive the adaptation of the gate, one has to perform the reconstruction with fewer than the total number of eigenvectors. That is, *the reconstruction has to be done from a subspace*. However, one can still utilize the full set of eigenvectors or principal components as the overall system output, as may be required by PCA.

4.4 The PDF of Reconstruction Error

For the MOE, it is necessary to specify $f(x - \tilde{x}_k) = p(d|x, k)$, the conditional pdf of the desired signal, given the input and the parameters of the k^{th} expert. For regression, $p(d|x, k)$ is usually modeled as a Gaussian distribution in the error between the desired signal and the k^{th} experts' output, which for auto-association becomes

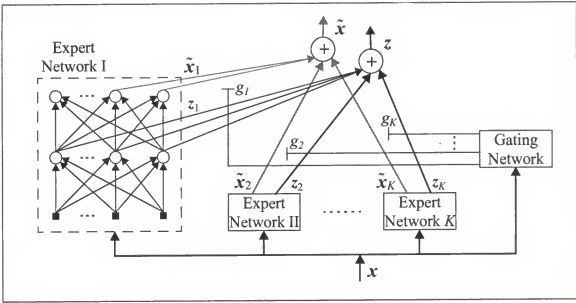


Figure 4-1. Competitive PCA network implemented using the Mixture of Experts architecture.

$$p(d|x, k) = \frac{1}{(2\pi)^{D/2} |\Sigma_k|^{1/2}} \exp \left[-\frac{1}{2} (x - \tilde{x}_k)^T \Sigma_k^{-1} (x - \tilde{x}_k) \right] \quad (4.10)$$

where Σ_k is the weighted covariance of the reconstruction error of the k^{th} expert. Equation (4.10) is perfectly valid when the experts are non-linear auto-associators. However, as we will now show, when the experts are linear, Σ_k is not full rank and thus not invertible. Consider a single PCA system with eigenvectors $U = [U_p \ U_s]$, where the columns of U have been separated into the principal and secondary eigenvectors, respectively. Noting that $UU^T = U_p U_p^T + U_s U_s^T = I$, the reconstruction error of a single input can be written as

$$x(n) - \tilde{x}(n) = x(n) - U_p U_p^T x(n) = [I - U_p U_p^T] x(n) = U_s U_s^T x(n) \quad (4.11)$$

and the covariance of the reconstruction error is given by

$$\Sigma = E[(x - \tilde{x})(x - \tilde{x})^T] = U_s U_s^T E[xx^T] U_s U_s^T = U_s U_s^T R U_s U_s^T = \sum_{i=P}^{D-1} \lambda_i u_i u_i^T. \quad (4.12)$$

Thus, the rank of the covariance matrix is given by the number of secondary eigenvectors, $S = D - P$. Since the covariance matrix is square and of dimension D , it is not full rank for any dimensionality reduction. Ultimately, all such direct attempts at formulating PCA in a statistical framework fail because *PCA is not a model for the data but merely a transformation of the data based on a decomposition of the covariance matrix*. This is an undesirable situation because we would like to use the same statistical framework for both linear and non-linear auto-association. One way around this is to borrow some results from factor analysis. Factor analysis is a *model* that attempts to explain an observed high dimensional random vector in terms of an unobserved lower dimensional random vector. We now show that, under certain simplifying assumptions, factor analysis defaults to a PCA expansion.

Here we examine the conditions under which factor analysis reduces to principal component analysis, for the purpose of finding a statistical framework for PCA. According to Morrison (1976), this special factor model was studied by Lawley (1953). Here we present an original derivation. We assume that an observed Gaussian random vector, $\mathbf{x} \sim N_D(\mathbf{0}, \Sigma)$, can be explained in terms of a postulated but unobserved lower dimensional Gaussian random vector, $\mathbf{z} \sim N_P(\mathbf{0}, \mathbf{I})$, such that

$$\mathbf{x}(n) = \mathbf{A}\mathbf{z}(n) + \mathbf{e}(n) \quad (4.13)$$

where \mathbf{A} is a $D \times P$ matrix, $D \geq P$, and \mathbf{e} is independent Gaussian noise, $\mathbf{e} \sim N_P(\mathbf{0}, \Psi)$, with Ψ assumed to be diagonal. Given these assumptions, then \mathbf{x} is also Gaussian distributed, $\mathbf{x} \sim N_D(\mathbf{0}, \Sigma)$, with a population covariance given by

$$\Sigma = E[\mathbf{x}\mathbf{x}^T] = \mathbf{A}\mathbf{A}^T + \Psi. \quad (4.14)$$

Note the similarity between (4.13) and the PCA expansion equation (4.4). However, (4.13) is a model for the observed data \mathbf{x} in terms of the latent variable \mathbf{z} , while in (4.4), \mathbf{z} is simply the principal components. The goal of factor analysis is to estimate \mathbf{A} and Σ . This can be done by maximizing the likelihood of the sample population of \mathbf{x} under the assumed model for the covariance. For a sample data set of N vectors, the resulting likelihood of the observed samples is

$$L = \prod_{n=1}^N p(\mathbf{x}(n)) = \prod_{n=1}^N \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \text{Exp} \left[-\frac{1}{2} \mathbf{x}^T(n) \Sigma^{-1} \mathbf{x}(n) \right] \quad (4.15)$$

and the negative log-likelihood, which is our criteria to be minimized, can be written

$$C = -\text{Log}[L] = \frac{N}{2} \text{Log}[|\Sigma|] + \frac{N}{2} \text{Tr}[\mathbf{R} \Sigma^{-1}] + \text{Constant} \quad (4.16)$$

where \mathbf{R} is the sample covariance of \mathbf{x}

$$\mathbf{R} = \frac{1}{N} \sum_{n=1}^N \mathbf{x}(n) \mathbf{x}^T(n). \quad (4.17)$$

The maximum likelihood solution for \mathbf{A} and Σ is found by taking the partial derivative of (4.16) with respect to those parameters, and setting the results equal to zero. However, this can only find the space spanned by \mathbf{A} , since any orthogonal rotation of \mathbf{z} still leads to the same value of Σ . An additional constraint is needed to find a unique solution for \mathbf{A} . Here, we impose two constraints that are much stricter than those normally imposed in factor analysis: namely, that the columns of $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_p]$ are orthogonal, and that the components of the noise have equal variance

$$\mathbf{a}_i^T \mathbf{a}_j = \|\mathbf{a}_i\|^2 \delta_{i,j} \quad (4.18)$$

$$\Psi = \sigma^2 \mathbf{I}. \quad (4.19)$$

With these two constraints, factor analysis reduces to PCA. We will now outline a proof of this. Although we could use the calculus of variations and append the constraints to the criteria, it is more instructive to use them explicitly. The postulated population covariance can be written

$$\Sigma = \sigma^2 \mathbf{I} + \sum_{i=1}^P \mathbf{a}_i \mathbf{a}_i^T. \quad (4.20)$$

The first P eigenvalues of Σ can be readily found by multiplying (4.20) by \mathbf{a}_j

$$\Sigma \mathbf{a}_j = \sigma^2 \mathbf{a}_j + \sum_{i=1}^P \mathbf{a}_i (\mathbf{a}_i^T \mathbf{a}_j) = \sigma^2 \mathbf{a}_j + \|\mathbf{a}_j\|^2 \mathbf{a}_j = (\sigma^2 + \|\mathbf{a}_j\|^2) \mathbf{a}_j \quad 1 \leq j \leq P. \quad (4.21)$$

Simple examination of (4.20) shows that the remaining $S = D - P$ secondary eigenvalues are all given by σ^2 . Since the determinant of a matrix is equal to the product of its eigenvalues, the determinant and log determinant of Σ are given by

$$|\Sigma| = \sigma^{2S} \prod_{i=1}^P (\sigma^2 + \|\mathbf{a}_i\|^2) \quad (4.22)$$

$$\log |\Sigma| = S \log \sigma^2 + \sum_{i=1}^P \log [\sigma^2 + \|\mathbf{a}_i\|^2]. \quad (4.23)$$

The constraints in (4.18) and (4.19) allow us to simplify a well known matrix inversion theorem

$$\Sigma^{-1} = \frac{\mathbf{I}}{\sigma^2} - \frac{\mathbf{A}}{\sigma^2} \left[\mathbf{I} + \frac{\mathbf{A}^T \mathbf{A}}{\sigma^2} \right]^{-1} \frac{\mathbf{A}^T}{\sigma^2} = \frac{1}{\sigma^2} \left[\mathbf{I} - \sum_{i=1}^P \frac{\mathbf{a}_i \mathbf{a}_i^T}{\sigma^2 + \|\mathbf{a}_i\|^2} \right]. \quad (4.24)$$

Using (4.23) and (4.24), we can re-write the criteria of (4.16), after absorbing constants, as

$$C = S \log \sigma^2 + \sum_{i=1}^P \log[\sigma^2 + \|a_i\|^2] + \frac{1}{\sigma^2} \left[\text{Tr}[R] - \sum_{i=1}^P \frac{\text{Tr}[R a_i a_i^T]}{\sigma^2 + \|a_i\|^2} \right]. \quad (4.25)$$

Note that direct application of the constraints has completely decoupled the columns of A in the criteria. Minimizing (4.25) with respect to a_k results in the equation

$$\frac{a_k}{\sigma^2 + \|a_k\|^2} + \frac{(a_k^T R a_k) a_k}{\sigma^2 (\sigma^2 + \|a_k\|^2)^2} - \frac{R a_k}{\sigma^2 (\sigma^2 + \|a_k\|^2)} = 0 \quad (4.26)$$

which can only be true in general if a_k is an eigenvector of R

$$R \hat{a}_k = \lambda_k \hat{a}_k \quad 1 \leq k \leq P. \quad (4.27)$$

We have thus shown that, under the assumptions of (4.18) and (4.19), factor analysis is equivalent to a PCA expansion. Substituting (4.27) back into (4.26) allows us to solve for the magnitudes of the eigenvectors

$$\|\hat{a}_k\|^2 = \lambda_k - \sigma^2 \quad 1 \leq k \leq P. \quad (4.28)$$

Substituting (4.28) into (4.25), and making use of the theorem that the trace of a matrix is equal to the sum of its eigenvalues, results, after simplification, in the reduced criterion

$$C = S \log \sigma^2 + \sum_{i=1}^P \log \lambda_i + P + \frac{1}{\sigma^2} \sum_{i=P+1}^D \lambda_i. \quad (4.29)$$

Minimizing (4.29) with respect to σ^2 immediately yields

$$\hat{\sigma}^2 = \frac{1}{S} \sum_{i=P+1}^D \lambda_i \quad (4.30)$$

which is the average value of the secondary eigenvalues. We have now fully determined the maximum likelihood solution for the free parameters of the model, \mathcal{A} and σ^2 . We can gain further insight into the solution by substituting (4.28) and (4.30) into the assumed model (4.20) for the covariance. If we represent the columns of \mathcal{A} in terms of their magnitude and a unit direction, $\mathbf{a}_i = \|\mathbf{a}_i\| \cdot \mathbf{u}_i$, where the \mathbf{u}_i are the normalized eigenvectors of \mathbf{R} , then the model covariance and its inverse evaluated at the maximum likelihood solution can be written

$$\hat{\Sigma} = \sum_{i=1}^P \lambda_i (\mathbf{u}_i \mathbf{u}_i^T) + \hat{\sigma}^2 \sum_{i=P+1}^D (\mathbf{u}_i \mathbf{u}_i^T) \quad (4.31)$$

$$\hat{\Sigma}^{-1} = \sum_{i=1}^P \frac{1}{\lambda_i} (\mathbf{u}_i \mathbf{u}_i^T) + \frac{1}{\hat{\sigma}^2} \sum_{i=P+1}^D (\mathbf{u}_i \mathbf{u}_i^T). \quad (4.32)$$

Both (4.31) and (4.32) are very similar to the spectral decomposition of \mathbf{R} and its inverse, respectively, except the smallest S eigenvalues have all been replaced by their average, $\hat{\sigma}^2$. Note that while (4.31) is dominated by the principal eigenvalues, (4.32) is dominated by the secondary eigenvalues, and thus we can approximate the maximum likelihood solution for the inverse covariance by the second term only

$$\hat{\Sigma}^{-1} \approx \frac{1}{\hat{\sigma}^2} \sum_{i=P+1}^D \mathbf{u}_i \mathbf{u}_i^T = \frac{1}{\hat{\sigma}^2} \mathbf{U}_s \mathbf{U}_s^T. \quad (4.33)$$

The Mahalanobis distance for a given input vector \mathbf{x} is then given by

$$\mathbf{x}^T \hat{\Sigma}^{-1} \mathbf{x} \approx \frac{1}{\hat{\sigma}^2} \mathbf{x}^T \mathbf{U}_s \mathbf{U}_s^T \mathbf{x}. \quad (4.34)$$

However, from (4.11), the magnitude squared of the PCA reconstruction error is given by

$$\|e_{pca}\|^2 = e_{pca}^T e_{pca} = \mathbf{x}^T \mathbf{U}_s \mathbf{U}_s^T \mathbf{U}_s \mathbf{U}_s^T \mathbf{x} = \mathbf{x}^T \mathbf{U}_s \mathbf{U}_s^T \mathbf{x}. \quad (4.35)$$

Substituting (4.35) into (4.34), we can write

$$\mathbf{x}^T \hat{\Sigma}^{-1} \mathbf{x} \approx \frac{\|e_{pca}\|^2}{\hat{\sigma}^2}. \quad (4.36)$$

Returning now to the case of multiple experts, (4.36) suggests that we can write the pdf of the reconstruction error of the k^{th} experts as

$$p(\mathbf{d}|\mathbf{x}, k) \approx \frac{1}{(2\pi\sigma_k^2)^{S/2}} \exp\left(-\frac{\|\mathbf{x} - \tilde{\mathbf{x}}_k\|^2}{2\sigma_k^2}\right) \quad (4.37)$$

where the variance is given by the mean value of the secondary eigenvalues

$$\sigma_k^2 = \frac{1}{S} \sum_{i=P}^{D-1} \lambda_{ki} = \frac{1}{S} E[\|\mathbf{x} - \tilde{\mathbf{x}}_k\|^2] \quad (4.38)$$

and λ_{ki} is the i^{th} eigenvalue of the k^{th} expert. Note that this is the same as if we set the covariance of the reconstruction error in (4.11) to have the diagonal form

$$\Sigma_k = \sigma_k^2 \mathbf{I} \quad (4.39)$$

and assume S degrees of freedom. Thus, using (4.39), we can use the same reconstruction error model independent of whether the experts are linear or non-linear. It is natural then to question the consequences of not using the exact pdf in (4.37). First, even the exact formulation makes a Gaussian assumption, which may be violated by the data itself. Second, any approximation in the conditional pdf's of the experts can be accommodated by a small change in the mixture coefficients. Third, as we shall soon see, the approximation leads to a weighted mean square error derivation of the linear PCA parameters, similar to the derivation for a single PCA system. Finally, the model works sufficiently well in practice.

4.5 Solution of Competitive PCA Decomposition

Having found the appropriate pdf for PCA, the negative log-likelihood cost function is

$$C = \sum_{n=1}^N \sum_{k=1}^K \left\{ -h_k(n) \log[g_k(n)] + \frac{1}{2} h_k(n) \left[\frac{\|x(n) - \tilde{x}_k(n)\|^2}{\sigma_k^2} + S \log \sigma_k^2 + S \log 2\pi \right] \right\}. \quad (4.40)$$

We will now show that the competitive PCA network will find the principal components of each stationary segment of the input signal. Since PCA is a linear transformation, we can globally minimize the second part of (4.40) with respect to the experts' weights. Since the experts have been decoupled in (4.40), we can minimize that part of the cost function with respect to each expert separately

$$C_k = \frac{1}{2} \sum_{n=1}^N h_k(n) \|x(n) - \tilde{x}_k(n)\|^2. \quad (4.41)$$

Let the weights of the k^{th} expert be given by some unknown $P \times D$ matrix W_k so that the reconstruction error is given by

$$x(n) - \tilde{x}_k(n) = (I - W_k W_k^T) x(n) + b_k \quad (4.42)$$

where we have included the unknown constant vector b_k to allow for the possibility that x is not zero-mean. Then the cost function of the k^{th} expert is

$$\begin{aligned} C_k &= \frac{1}{2} \sum_{n=1}^N h_k(n) \{ [x^T(n)(I - W_k W_k^T) + b_k^T] [(I - W_k W_k^T)x(n) + b_k] \} \\ &= \frac{1}{2} \sum_{n=1}^N h_k(n) \{ x^T(n)(I - W_k W_k^T)x(n) + 2x^T(n)(I - W_k W_k^T)b_k + b_k^T b_k \} \end{aligned} \quad (4.43)$$

where the orthogonality condition, $W_k^T W_k = I$, allowed the simplification $(I - W_k W_k^T)^2 = (I - W_k W_k^T)$. Finding the constant b_k first:

$$\frac{\partial C_k}{\partial \mathbf{b}_k} = 0 \Rightarrow \mathbf{b}_k = -(\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T) \bar{\mathbf{x}}_k \quad (4.44)$$

$$\bar{\mathbf{x}}_k \equiv \frac{\sum_{n=1}^N h_k(n) \mathbf{x}(n)}{\sum_{n=1}^N h_k(n)}. \quad (4.45)$$

We can now re-write the cost function as

$$\begin{aligned} C_k &= \frac{1}{2} \sum_{n=1}^N h_k(n) \{ (\mathbf{x}(n) - \bar{\mathbf{x}}_k)^T (\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T) (\mathbf{x}(n) - \bar{\mathbf{x}}_k) \} \\ &= \text{Constant} \cdot \frac{1}{2} \text{Tr}[(\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T) \mathbf{R}_k] \end{aligned} \quad (4.46)$$

where \mathbf{R}_k is a weighted autocorrelation matrix

$$\mathbf{R}_k \equiv \frac{\sum_{n=1}^N h_k(n) [\mathbf{x}(n) - \bar{\mathbf{x}}_k][\mathbf{x}(n) - \bar{\mathbf{x}}_k]^T}{\sum_{n=1}^N h_k(n)}. \quad (4.47)$$

Let the i^{th} column of \mathbf{W}_k be given by \mathbf{w}_{ki} . Since the columns of \mathbf{W}_k are already decoupled in (4.46), we need only append a normality constraint to the cost function, which becomes, after absorbing constants

$$C_k = \frac{1}{2} \text{Tr}[\mathbf{R}_k] - \frac{1}{2} \sum_{i=1}^P \text{Tr}[\mathbf{w}_{ki} \mathbf{w}_{ki}^T \mathbf{R}_k] + \frac{1}{2} \sum_{i=1}^P \lambda_{ki} (\mathbf{w}_{ki}^T \mathbf{w}_{ki} - 1). \quad (4.48)$$

The minimum of (4.48) occurs when the \mathbf{w}_{kj} are chosen to be eigenvectors of \mathbf{R}_k

$$\frac{\partial C_k}{\partial \mathbf{w}_{kj}} = 0 \Rightarrow \mathbf{R}_k \mathbf{w}_{kj} = \lambda_{kj} \mathbf{w}_{kj} \quad (4.49)$$

but chosen so that the eigenvectors correspond to the largest eigenvalues. That is to say, for the optimal reconstruction of the k^{th} expert at each M step, we should choose the P largest eigenvectors of \mathbf{R}_k . *This clearly shows that the competitive PCA architecture is still solving an eigenvalue problem.* Furthermore, if the posterior probabilities converge to binary values indicating the stationary regions of the time series, then each experts' autocorrelation matrix and eigenvectors will correspond to a different stationary regime.

Finally, at each iteration, the variance of each experts' pdf is set to the average of the discarded eigenvalues, according to (4.38). We now summarize the algorithm in pseudo-code.

For each iteration, i :

E-Step:

For each vector in the data set, $n = 1 \dots N$

1. Calculate the gate outputs: $g_k(\mathbf{x}(n))$.

For each expert, $k = 1 \dots K$

2. Calculate the reconstruction estimate: $\tilde{\mathbf{x}}_k(n) = \mathbf{W}_k \mathbf{W}_k^T \mathbf{x}(n) + (\mathbf{I} - \mathbf{W}_k \mathbf{W}_k^T) \tilde{\mathbf{x}}_k$.

3. Evaluate the pdf of each expert: $p(\mathbf{d}(n)|\mathbf{x}(n), k) = \frac{1}{(2\pi\sigma_k^2)^{S/2}} \exp\left[-\frac{\|\mathbf{x}(n) - \tilde{\mathbf{x}}_k(n)\|^2}{2\sigma_k^2}\right]$.

4. Calculate the posterior probabilities: $h_k(n) = \frac{g_k(n)p(\mathbf{d}(n)|\mathbf{x}(n), k)}{\sum_{j=1}^K g_j(n)p(\mathbf{d}(n)|\mathbf{x}(n), k)}$

M-Step:

5. Train the gate using the posterior probabilities, $h_k(n)$, as targets at the input side of the soft-max output layer.

For each expert, $k = 1 \dots K$

6. Calculate the conditional means: $\tilde{\mathbf{x}}_k = \left[\sum_{n=1}^N h_k(n) \right]^{-1} \sum_{n=1}^N h_k(n) \mathbf{x}(n)$.

7. Calculate the autocorrelation matrices: $\mathbf{R}_k = \left[\sum_{n=1}^N h_k(n) \right]^{-1} \sum_{n=1}^N h_k(n) [\mathbf{x}(n) - \tilde{\mathbf{x}}_k][\mathbf{x}(n) - \tilde{\mathbf{x}}_k]^T$.

8. Calculate the eigenvectors, $j = 1 \dots P$: $R_k w_{kj} = \lambda_{kj} w_{kj}$.

9. Calculate the variance: $\sigma_k^2 = \frac{1}{S} \sum_{i=P}^{D-1} \lambda_{ki}$

The overall output, as defined by (4.9), can be calculated after the algorithm has converged.

4.6 Practical Implementation Issues

4.6.1 Applicability

We envision three primary applications of competitive Temporal PCA: signal segmentation, noise reduction, and adaptive encoding.

4.6.2 Network design

In designing a system, the following parameters must be determined: The number of experts, the number of principal components per expert, and the number of processing elements of the gating network.

The number of experts should be chosen so as to match the number of stationary regimes. In some cases, such as with speech, the number of stationary regimes can be estimated fairly well beforehand. In most cases, however, the number is unknown. In these cases, pruning or growing algorithms can be employed, but are beyond the scope of this paper.

The number of principal components required per expert should be chosen on the basis of the number of experts. With just a few experts, the first two or four principal components will almost always provide sufficient differentiation between regimes. Large numbers of experts may require more principal components.

Spurious switching can be influenced by the size of the gating network. Recall that the posterior probabilities act as a target for the gate. If the gate has too many free parameters,

it will tend to overfit the posterior probabilities, which can exhibit spurious switching errors due to noise. In this case, the gate will simply learn the switching errors. On the other hand, a gating network with a minimal number of free parameters is less likely to memorize spurious switching errors, and thus is more likely to provide a smoother segmentation. Thus, it is important to find the gating network with the minimal number of free weights that can still learn the posterior targets reasonably well.

4.6.3 Initialization

The weights of the gating network should be initialized to small values so that the gate estimates all experts as being nearly equiprobable over the entire data set. The expert variances should be initialized to large values, to express uncertainty in the initial weights of the PCA experts.

There are several different ways to initialize the PCA experts. From the global KLT for the entire data set, the global eigenfilters can be calculated. In one scenario, all the expert eigenfilters can be made equal to the global eigenfilters, plus small random perturbations. In practice, this is equivalent to a completely random initialization. This is because early in the training, the gate estimates all experts as being equally probable, and since the expert variances are initially large, the experts' posterior probabilities are nearly identical over the entire data set. Thus, at least after the first iteration through the data set, the experts approximate the global eigenfilters. Another initialization technique, which we prefer, is to assign the 1st and 2nd global eigenfilters to the first expert, the 3rd and 4th global eigenfilters to the second expert, etc. The remaining eigenfilters of each expert are then initialized to normalized random vectors.

4.6.4 Training Issues

In our experience, the pdf's dominate the calculation of the posterior probabilities. The gate's contribution to the posterior probabilities during training is primarily to reinforce the pdf's, by learning the posterior probabilities, and to interpolate at the boundaries between regimes. Therefore, at a particular training iteration, it is not necessary to train the gate to full parameter convergence.

For some harder problems, we find that annealing the number of principal components, starting from $P = 2$ and progressing up to the final dimensionality reduction, yields better results.

4.6.5 Repeatability and convergence

We wish to emphasize that not all training runs converge to a reasonable solution. However, degenerate solutions are usually quite obvious, often resulting in a very rapid switching between experts every few samples, with no visible segmentation. Among runs that converge, we find that they produce qualitatively similar segmentations. However, there can be differences in the exact location of major regime transitions and in the amount of spurious switching. As with all complex adaptive systems, training must be repeated several times and the results compared. However, since the algorithm is unsupervised, care must be exercised in choosing the best training.

4.7 Extensions

4.7.1 Minimum eigenvalue approach

Pisarenko's (1973) harmonic retrieval (PHR) method is an eigenanalysis technique for the special case when a time series consists of superimposed sinusoids in additive noise. It can be shown that, for a given eigenvector u_i with associated eigenvalue, λ_i , all eigenvec-

tors u_j with associated eigenvalue λ_j , where $j > i$, will have a zero at the location of the peak of the spectrum of u_i . In practice, this also holds true for more complex signals. Thus, because of the orthogonality constraint, the eigenvector associated with the smallest eigenvalue essentially contains information about all the other eigenvectors. The MUSIC algorithm exploits this by looking at the inverse spectrum of the eigenfilters in the noise subspace (those with the smallest eigenvalues).

Inspired by Pisarenko's work, instead of using the experts' reconstruction error as the competitive basis, we seek a cost function that is minimum for an orthogonal projection between the input and a weight vector. Thus, if the output of a single expert is

$$y(n) = \mathbf{w}^T \mathbf{x}(n) \quad (4.50)$$

then y^2 would be the competitive metric. In the general case, minimizing y^2 over the entire data set, subject to the constraint $\mathbf{w}^T \mathbf{w} = 1$, is equivalent to finding the eigenfilter associated with the smallest eigenvalue

$$\text{Min}_{\mathbf{w}} \{E[y^2]\} = \text{Min}_{\mathbf{w}} \{E[\mathbf{w}^T \mathbf{R} \mathbf{w}]\} \Rightarrow \mathbf{R} \mathbf{w} = \lambda_{\min} \mathbf{w}. \quad (4.51)$$

While the maximum eigenfilter can be regarded as a matched bandpass filter, the minimal eigenfilter can be regarded as an inverse filter. The minimal eigenfilter can also be found adaptively on-line through anti-Hebbian learning. In all cases, integration with the Mixture of Experts formalism is straightforward.

4.7.2 Modeling Power Variations Within a Regime

It is often desirable to recognize a particular stationary regime independent of its power. Although matched eigenvectors are invariant to changes in the power of a signal, the eigenvalues are not. Since the variance is equal to the mean of the discarded eigenvalues, power variations can affect the expert pdf's and posterior probabilities. A single vari-

ance estimate for each of the experts may be inadequate in this case. An alternative approach is to model the variance as a function of the input: $\sigma_k = \sigma_k(\mathbf{x})$. This can be accomplished with a neural network that learns to predict the variance on a sample by sample basis (Bishop, 1995).

4.7.3 Principal Sub-space Decomposition

One extension that has already been mentioned is to use a principal sub-space decomposition, instead of a principal component decomposition. This may speed up on-line learning, since it is no longer necessary that the eigenvectors converge in sequence, as is required for the on-line PCA algorithms. Principal sub-space decomposition also allows the use of non-linear expert networks.

4.7.4 Reducing Spurious Gate Switching

If signal segmentation is the primary goal, then it is desirable to remove spurious gate switching. One way is to accomplish this is to low pass “filter” the posterior probabilities, by computing the joint posterior probabilities for neighboring time steps, and then use these as the target for the gate. A second way to remove spurious gate switching is to add some form of memory to the gating network. In the context of competitive prediction, Weigend (1995a) used both global feedback of the gate output to the input, as well as an exponential memory on the hidden units of the gating network, and found that the latter approach worked better.

4.7.5 On-line learning

When gradient descent on the global cost function (4.40) is used for adaptation, all the free parameters of the competitive PCA architecture can be updated on a sample by sam-

ple basis. Sanger's rule computes the principal components in an on-line approach. The matrix form of Sanger's rule

$$\Delta W(n) \propto y(n)x^T - LT[y(n)y^T(n)]W(n) \quad (4.52)$$

in the MOE framework becomes

$$\Delta W_k(n) \propto \frac{h_k(n)}{\sigma_k^2} \{y_k(n)x^T - LT[y_k(n)y_k^T(n)]W_k(n)\} \quad (4.53)$$

where LT denotes the lower triangular operator, which sets all elements of the matrix above the diagonal equal to zero. The learning rates of the gate and of the expert networks must be carefully coordinated.

4.8 Application to Images

4.8.1 Texture Segmentation

We applied the model to a 504x504 grayscale image consisting of five textures, shown in Figure 4-2. The training set consisted of 1,764 non-overlapping sub-blocks of 12x12 pixels. There were five linear PCA experts, each performing a 144 to 16 dimensionality reduction (and a 16 to 144 expansion for reconstruction). The gating network was a two hidden layer multi-layer perceptron with 15 and 10 hyperbolic tangent activation functions on the hidden layers, respectively, resulting in an overall architecture of 144-15-10-5.

The output of the gate after training is shown in Figure 4-3 and the posterior probabilities, which act as a target for the gate, are shown in Figure 4-4. For each figure, panel (f) represents the winner-take-all segmentation, grayscale coded to indicate the winning expert for each sub-block. From these, we see that the image has been fairly successfully

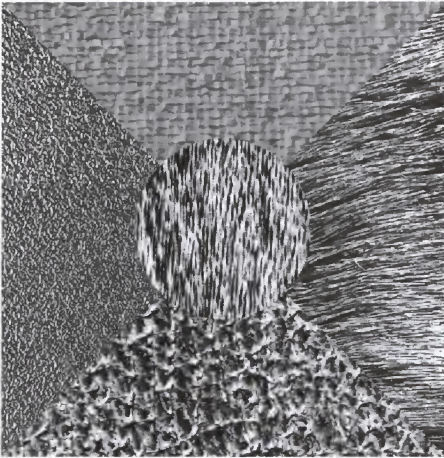


Figure 4-2. Training image.

segmented, with most of the errors occurring in the south texture. Also, the gate has learned the posterior probabilities very well.

Figure 4-5 shows the masks (the PCA weights arranged as a matrix and gray scale coded as to value) of the five experts. The experts are ordered from top (expert 1) to bottom (expert 5), and the masks are ordered from the largest (left) to smallest (right) corresponding eigenvalues. Only the first eight of the sixteen masks are shown. We see that each expert clearly specialized on a particular feature of a specific texture.

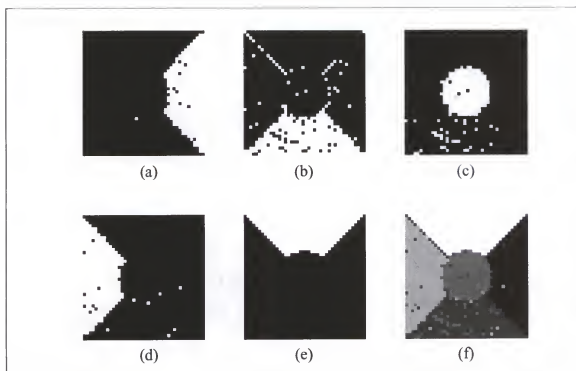


Figure 4-3. Training image gate segmentation: (a)-(e) gate outputs 1-5 corresponding to experts 1-5, respectively; (f) winner take-all segmentation.

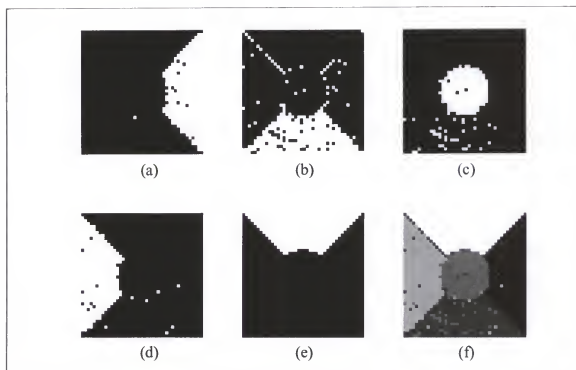


Figure 4-4. Training image posterior probabilities segmentation: (a)-(e) gate outputs 1-5 corresponding to experts 1-5, respectively; (f) winner take-all segmentation.

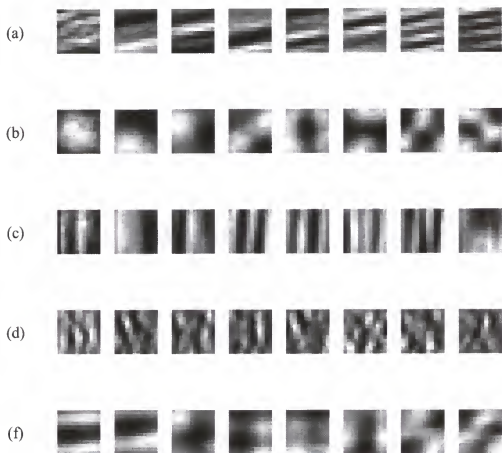


Figure 4-5. (a)-(f) First eight PCA masks for experts 1-5, respectively.

We tested the trained model on the 504x504 grayscale image shown in Figure 4-6. It consists of five patterns, two of which were also present in the training image, and one of which is very similar to a texture in the training image.

The testing results for the gate and posterior probabilities are shown in Figure 4-7 and Figure 4-8, respectively. We see that the gate has done only a fair job of segmenting the test image, with expert 1 fully recognizing the movement of the north texture in the training image to the east texture in the testing image, but only partially recognizing the movement of the west texture in the training image to the north position in the testing image.

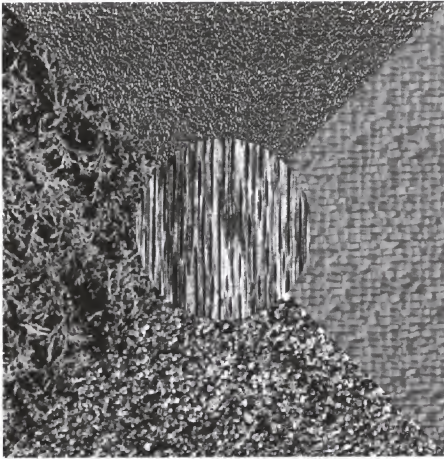


Figure 4-6. Testing image.

However, the posterior probabilities demonstrate a much better segmentation. Even the center textures, which are similar but not completely equivalent between the training and test images, were recognized as belonging to the same texture class.

4.8.2 Real Image Segmentation

We then applied the model to a 560x704 grayscale image of a real world scene, shown in Figure 4-9, consisting of a sloped residential street in the foreground with rolling hills and clear sky in the background. The training set consisted of 6,160 non-overlapping sub-blocks of 12x12 pixels. There were eight linear PCA experts, each performing a 64 to 16

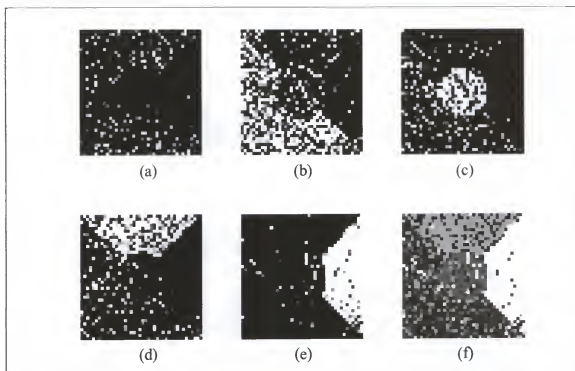


Figure 4-7. Test image gate segmentation: (a-e) gate outputs 1-5 corresponding to experts 1-5, respectively; (f) winner take-all segmentation.

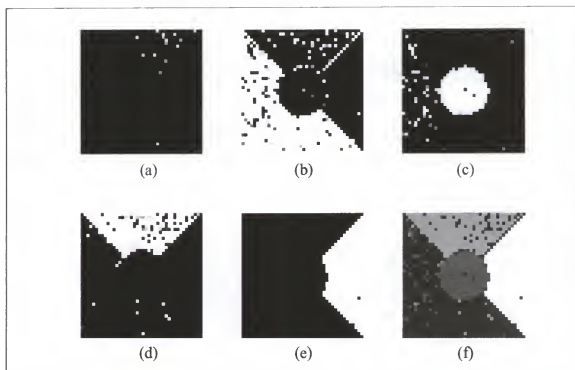


Figure 4-8. Test image posterior probabilities segmentation: (a-e) posterior probabilities corresponding to experts 1-5, respectively; (f) winner take-all segmentation.



Figure 4-9. Real world training image.

dimensionality reduction (and a 16 to 64 expansion for reconstruction). The gating network was a two hidden layer multi-layer perceptron with 48 and 32 hyperbolic tangent activation functions on the hidden layers, respectively, resulting in an overall architecture of 64-48-32-8.

The results after training for the gate and posterior probabilities are shown in Figure 4-10 and Figure 4-11, respectively. From the figures, we see that the PCA experts have begun to specialize on different textures within the image. For example, expert one (a) has

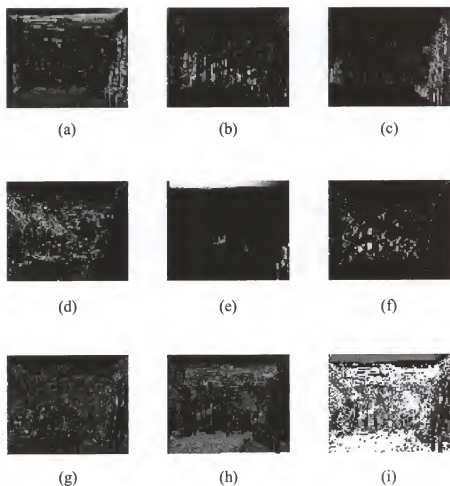


Figure 4-10. Real world image gate segmentation: (a-h) gate outputs 1-8 corresponding to experts 1-8, respectively; (i) winner take-all segmentation

specialized on the distant fields near the horizon, expert four (d) on the roofs of the houses, expert five (e) on the sky, and expert six (f) on certain windows.

It is also apparent that the gate has not converged to the posterior probabilities, in spite of the gate having almost 4,900 adaptable weights. Clearly this is difficult problem for a neural network to learn. Still, the gate provides a critical smoothing function.

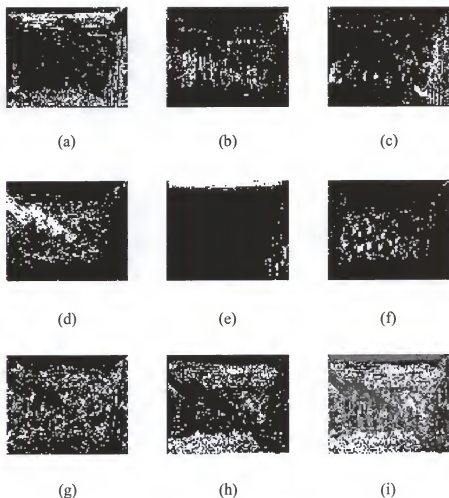


Figure 4-11. Real world image posterior probabilities segmentation: (a-h) gate outputs 1-8 corresponding to experts 1-8, respectively; (i) winner take-all segmentation

4.9 Conclusion

It is clear that competitive PCA is not only doing data reduction but also segmentation. This is an important observation since conventional PCA treats the entire image as a unique class, i.e. it is limited to the representation of the image. With our scheme each PCA component can represent a subset of the image patterns. Competition is utilized to find each piece, and thus the method brings discrimination into PCA analysis, which has not been done in the past.

CHAPTER 5 TEMPORAL PRINCIPAL COMPONENT ANALYSIS

5.1 Application of Competitive PCA to Time Series

We now consider the application of competitive PCA to the case when the system input is a time series fed through a tapped delay line. The architecture of each expert is then as shown in Figure 5-1, which we call temporal PCA.

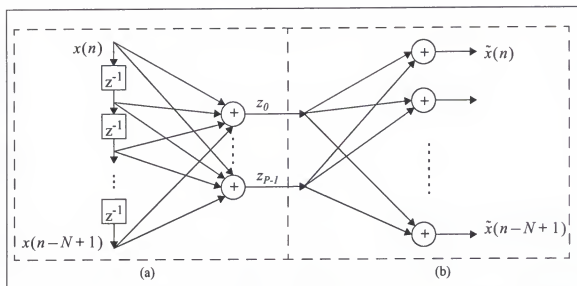


Figure 5-1. Temporal PCA architecture: (a) PCA network; (b) reconstruction network.

In this case, the autocorrelation matrix, \mathbf{R} , of dimension N , is a Toeplitz matrix with components given by the temporal autocorrelation function, $R_{mn} = r(m-n)$. The solutions to the eigenvalue equation, $\mathbf{R}\mathbf{u} = \lambda\mathbf{u}$, are called eigenfilters, since they act as time domain filters for the time series.

5.2 Competitive PCA Simulations

We now present three simulations to demonstrate the properties of the algorithm; one with artificially created data, and the others with real-world data.

5.2.1 Signal Segmentation: Switching FIR Process

We first tested the algorithm on a toy problem consisting of a switching FIR process. A time series was generated by a Gaussian noise driven FIR filter with 20 taps. Every 100 samples, the filter's coefficients were switched between one of two normalized weight sets. The total time series generated was 400 samples long. The resulting time series is shown in Figure 5-2.a, along with lines indicating the segmentation. The time series is difficult to segment by eye. Depending on the local properties of the driving noise, different spectral peaks can be excited within the same stationary region. There were two experts, each performing a dimensionality reduction of 20:6. The gate was a TDNN with two hidden layers and architecture 20-4-3-2. The gate and experts were trained for 200 epochs of the EM algorithm. For each M step, the gate was trained for 10 epochs. A reasonable segmentation was achieved after only 25 EM iterations, but with additional training the transitions became better defined and spurious gate switching errors were reduced. The results after training, shown in Figure 5-2., show that the gate has successfully segmented the switching FIR process. There is spurious switching near the transitions, and especially during the stationary regime from samples 200 to 300. However, such switching is brief and represents a small fraction of the total time. Figure 5-3 clearly demonstrates the tendency of the matched eigenfilters to pick out peaks in the signal spectrum. Examining column (b), the largest peak of FIR system II occurs at approximately 0.4 (Nyquist = 1), with a slightly smaller peak at 0.85. The spectra of the first two eigenfilters of expert II are

matched to the largest peak, effectively ignoring the slightly smaller peak. The spectra of the 3rd and 4th eigenfilters are matched to the 2nd largest peak only, and ignore the largest peak. The same behavior is observed for expert I.

Figure 5-4 shows various convergence metrics versus the number of EM iterations. This is important because we mathematically showed that each PCA expert develops a model for the individual time series segment provided the segmentation is correct. In practice, however, the segmentation will always be noisy. Therefore, an important question is to find out experimentally how fast the PCA experts converge to the true eigenvectors as a function of the quality of the segmentation. A segmentation quality metric for an expert was defined as the mean value of either the gate output or posterior probabilities during the regime it was trying to model. This metric has a value of one for perfect segmentation, and 0.5 for a segmentation chosen randomly. Panels (a) and (b) show the evolution of the segmentation quality of the experts during training, as measured by the gate and posterior probabilities. Convergence is monotonic, with the gate trailing the posterior probabilities, which have nearly converged by the 25th iteration. This is to be expected since the posterior probabilities act as targets for the gate. Panel (c) shows the evolution of the angle between each expert's 1st eigenvector and the true 1st eigenvector for the regime the expert is trying to model. We see that by the 25th iteration, the experts' eigenvectors have nearly converged to the true eigenvectors, even though the segmentation is only approximately 90% accurate, which shows that accurate models can be obtained in practice.

5.2.2 Signal Segmentation: Earthquake Seismogram

This experiment was performed with a real world data set consisting of a seismogram of a magnitude 6.8 earthquake that occurred near Jalisco, Mexico on May 1, 1997. The

data is the vertical long-period ($f_s=1$ second) seismogram recorded by the “NEW” station, and is shown in panel (a) of Figure 5-5. In the same graph, the vertical lines represent human expert scoring¹ of phase transitions between various direct and reflected primary, secondary, and surface waves of the earthquake.

There were five experts, each performing a dimensionality reduction of 60:4. The gate was a TDNN with two hidden layers and architecture 20-9-7-5. The gate and experts were trained for 1000 epochs of the EM algorithm. For each M step, the gate was trained for 10 epochs. The five gate outputs are also shown in Figure 5-5. We see that different experts have specialized on different temporal regions of the time series, and that there is some correlation between the expert segmentation and the gate segmentation. We did not, however, attempt to compare the gate output with the human expert identification of the segments.

5.2.3 Noise Reduction

Consider the case where a time series $x(n)$ is corrupted by additive zero-mean white noise. Let $v(n)$ be the additive noise and let the vector \mathbf{w} be some arbitrary FIR filter. The signal to noise power ratio at the output of the filter is given by $SNR = E[\mathbf{w}^T \mathbf{R} \mathbf{w}] / (\sigma^2 E[\mathbf{w}^T \mathbf{w}])$, where σ^2 is the noise power. It is not difficult to show the well known result that maximizing the signal to noise ratio implies that \mathbf{w} is an eigenvector of the autocorrelation matrix: $Max_{\mathbf{w}}[SNR] \Rightarrow \mathbf{R} \mathbf{w} = \lambda \mathbf{w}$. By symmetry arguments, the SNR will also be maximized in the reconstructed signal. One can exploit this property to filter out additive noise from time series, assuming that the time series is stationary. If it is not, the reconstructed signal can show dramatic distortion. We can achieve

1. Dr. Doug Smith of the University of Florida Geology Dept. provided the expert scoring.

further noise reduction, through a technique given by Danilov et al. (1996), by noting that, at the final reconstructed output vector, the components $\tilde{x}_i(n)$ and $\tilde{x}_{i+j}(n+j)$ both represent estimates of the same point in the time series, $x(n-i)$. Thus, we can improve the reconstruction estimate by forming the average:

$$\tilde{x}_{ave}(n) = \frac{1}{P} \sum_{i=0}^{P-1} \tilde{x}_i(n+i) \quad (5.1)$$

We first tested the noise reduction properties of the algorithm on a chirp signal with -6 dB additive white noise. The chirp illustrates our point very well and is the limiting case of a switching process so our methodology can still be applied. There were 4 experts, each performing a dimensionality reduction of 24 to 2. The gate was a TDNN with two hidden layers and architecture 24-6-5-4. The gate and experts were trained for 200 epochs of the EM algorithm. For each M step, the gate was trained for 10 epochs. A reasonable segmentation was achieved after only 25 EM epochs, followed by minor improvements. The results after training are shown in Figure 5-6. Panel (d) shows that the gate has developed an intuitive segmentation, with each expert specializing on a different region of the chirp signal. There is some minor spurious switching around transitions between the regimes defined by the gate. However, this is to be expected, since a chirp signal violates the assumption of piecewise stationarity. The frequency spectra of the first principal eigenvector of the four experts are shown in Figure 5-7. The graphs clearly show that each expert is specializing on a different region of the chirp signal in the frequency domain. We can quantitatively compare the competitive and global PCA techniques by calculating the mean square error between the original signal and the reconstructions. The results are

shown in Table 5-1. Competitive PCA has a smaller reconstruction than the global PCA reconstruction, independent of the dimensionality reduction, P .

Table 5-1. Mean square reconstruction error as a function of the number of principal components, P .

Competitive PCA $P=2$ per Expert	Global KLT		
	$P=4$	$P=8$	$P=12$
0.034	0.060	0.047	0.081

The reconstructed waveforms are plotted in Figure 5-8. Competitive PCA clearly provides a better reconstruction than the global PCA reconstructions. Furthermore, the global PCA reconstructions demonstrate a trade-off between the number of principal components and the quality of the reconstruction. If too few principal components are used, then the reconstruction exhibits amplitude modulation, due to the bandpass nature of the optimal eigenfilter. If too many principal components are used, then the reconstruction matrix approaches the identity matrix, and both signal and noise are reconstructed. Competitive PCA is able to deal independently with the noise reduction and the modeling.

We then tested the noise reduction properties of the algorithm on a digital recording ($f_s = 11127$ Hz) of a violin playing four notes: C-A#-A-A#, where A corresponded to standard concert pitch (440 Hz). Observe that the 2nd and 4th notes are nominally the same pitch. Since the violin has no frets, two nominally identical notes can still differ slightly in fundamental frequency, and transitions between notes may occur gradually. To the digital recording, we added white Gaussian noise for a signal to noise ratio of 0 dB, which then became the input to our algorithm. There were 3 experts, each fed by a tap-delay line with 100 taps. The gate was a TDNN with two hidden layers and architecture 100-6-5-3.

Because of the size of the data set, we used a slightly different training procedure. Each iteration corresponded to 5 epochs for training the experts and then 200 epochs for training the gate. The initial number of principal components was $P = 2$, which was annealed up to $P = 10$, by adding 2 principal components at each iteration, for a total of 5 iterations. A reasonable segmentation was achieved after only 2 iterations.

Figure 5-9 shows the results after training. Qualitatively, we see in panel (c) that the reconstructed time series has removed much of the noise while retaining the envelope of the original signal. To the ear, there is a vast reduction in the noise level, but tonal quality was changed such that the signal was not readily identifiable as a violin by unbiased listeners. By subtracting the original signal from the reconstructed version, we estimated a signal to noise ratio in the reconstructed signal of 47 dB. Panels (d), (e) and (f) show the gate output for the 3 experts and shows that the gate successfully segmented the signal. Furthermore, the gate output corresponding to the 2nd expert properly indicates that the 2nd and 4th notes are the same. This gives us confidence that the experts' eigenfilters do indeed represent the stationary regimes of the time series. Figure 5-10 compares the periodogram of the stationary regions of the original signal, as determined by the gate segmentation, with the corresponding experts' eigenfilters frequency response. Only the 1st, 3rd, 5th, 7th, and 9th eigenfilters are shown, since the even eigenfilters' frequency response are nearly identical to the odd ones. Each expert's 1st eigenfilter is centered around one of the fundamental frequencies of the three pitches present in the note sequence. This is in spite of the fact that the eigenfilters have a bandwidth, D^{-1} , of approximately 110 Hz, which is larger than the 83 Hz difference between the A and C notes.

Table 5-2. Peak frequencies (Hz) of the signal periodogram.

Regime corresponding to note	Harmonic				
	Funda- mental	1st	2nd	3rd	4th
C	535.3	1070.6	1599.9	2136.7	2666.6
A#	469.1	938.3	1408.9	1878.1	2347.2
A	445.1	887.2	1333.7	1777.3	2222.4

Table 5-3. Peak frequencies (Hz) of the expert and global eigenfilters.

Expert corresponding to note	Eigenfilter Order				
	1st	3rd	5th	7th	9th
C	534.1	1073.8	2136.4	1607.9	467.3 612.0
A#	467.3	2331.1 2815.1	2342.2 2809.6	929.1	1869.3
A	450.6	1774.8	1335.2	3115.6	3555.1
Global	489.6	417.3 567.5	2331.1 2820.7	2342.2 2809.6	923.54 1079.3

Table 5-2 shows the peak frequencies of the original time series as measured from the global periodogram. Table 5-3 shows the peak frequencies of the expert and global eigenfilters for the noisy time series. By comparing the two tables, we see that the peak frequencies of the 1st eigenfilters of the experts do indeed correspond to the fundamental frequencies of the three notes. The double entries in some of the cells indicate that some of the eigenfilters exhibit two peaks of comparable power. For example, the 9th eigenfilter of the note C expert has two peaks at 467.3 and 612.0 Hz. This split is not caused by interference from the other fundamentals, since the gate segmentation is nearly perfect. Rather, it is caused by residual energy around the note C fundamental that the 1st eigenfilter did not match. The splitting occurs because the orthogonality constraint means that higher order

eigenfilters have zeros near the peak locations of lower order eigenfilters. The global eigenfilters, on the other hand, do exhibit interference across stationary regions. For example, the 1st global eigenfilter has a peak at 489.6 Hz, which is close to the average of the three fundamental peaks.

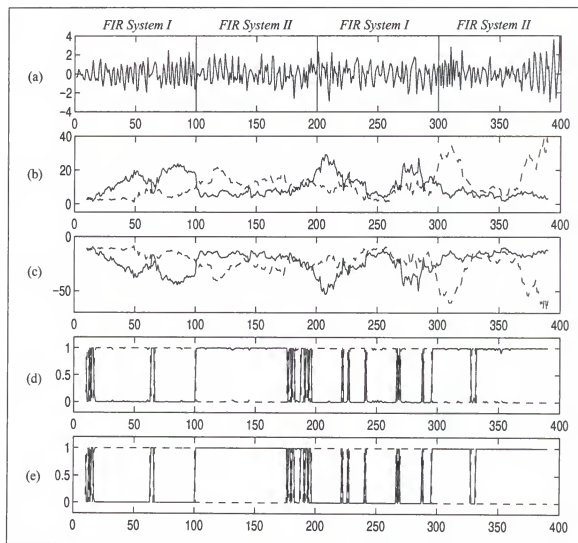


Figure 5-2. Switching regime simulation: (a) switching FIR signal; (b) expert squared reconstruction error; (c) expert log probability; (d) gate output; (e) posterior probability.

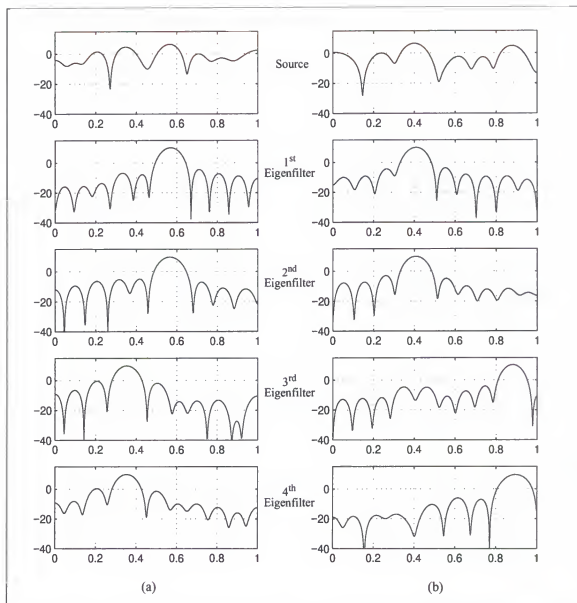


Figure 5-3. Switching regime simulation. Power spectra (dB) of source and expert matched eigenfilters: (a) source I and expert I eigenfilters; (b) source II and expert II eigenfilters.

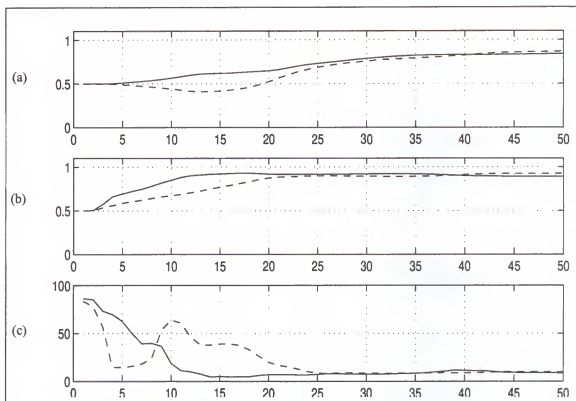


Figure 5-4. Switching regime simulation. Convergence properties for expert I (solid) and expert II (dashed): (a) gate segmentation quality vs. iteration number; (b) posterior probabilities segmentation quality vs. iteration number; (c) angle (degrees) between expert's 1st eigenvector and true 1st eigenvector vs. iteration number.

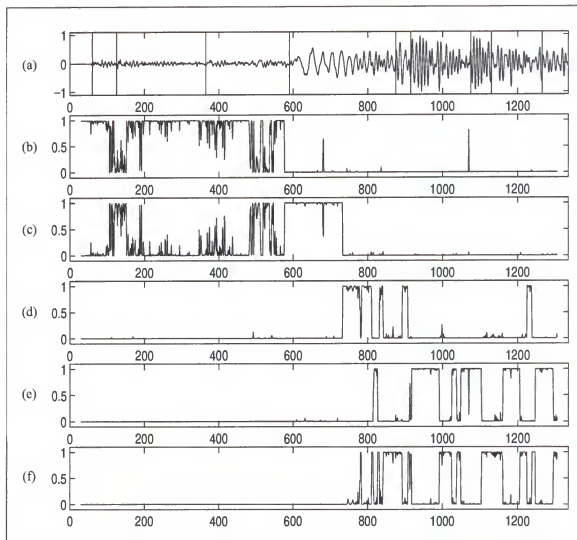


Figure 5-5. Seismogram segmentation: (a) seismogram and human expert scoring; (b),(c),(d),(e), and (f) gate outputs of experts 1,2,3,4, and 5, respectively

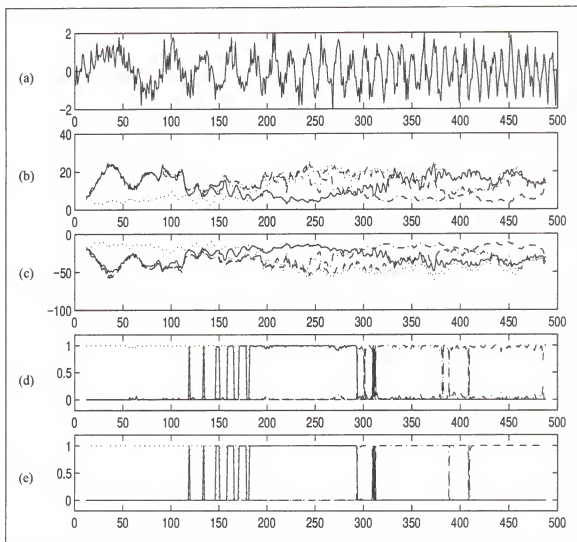


Figure 5-6. Chirp simulation: (a) corrupted signal; (b) expert squared reconstruction error; (c) expert log probability; (d) gate output; (e) posterior probability.

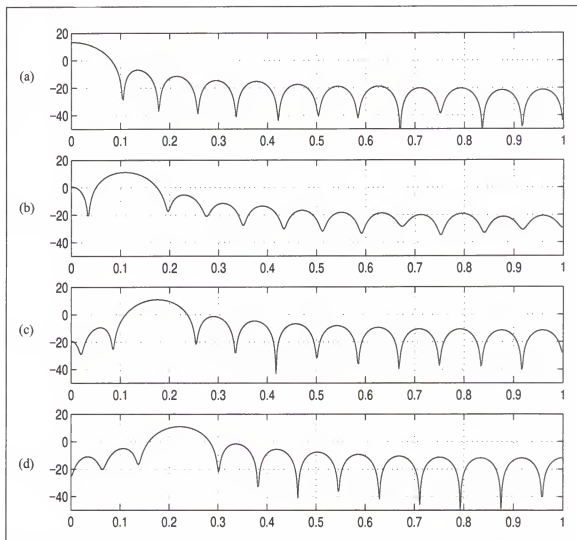


Figure 5-7. Chirp simulation: (a,b,c,d) frequency response of 1st principal eigenfilter for experts 1,2,3, and 4, respectively.

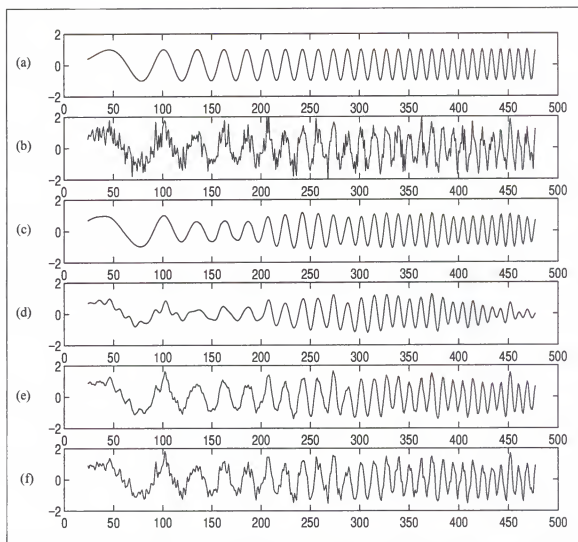


Figure 5-8. Chirp simulation: (a) original signal; (b) corrupted signal; (c) competitive PCA reconstruction; (d,e,f) global KLT reconstruction $P=4,8,12$.

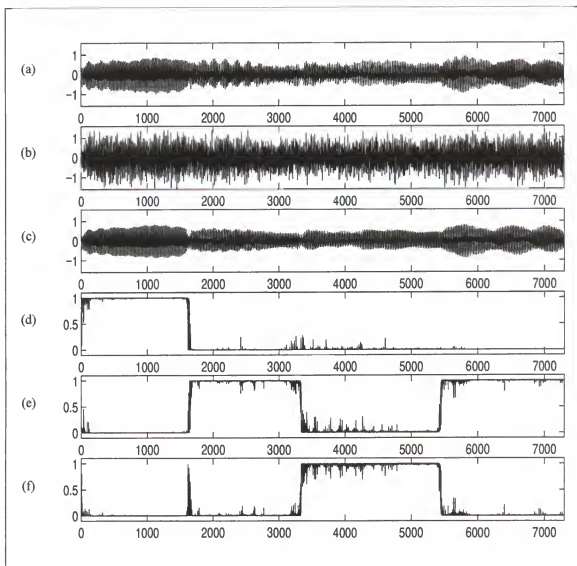


Figure 5-9. Violin: (a) original signal; (b) corrupted signal; (c) competitive PCA reconstruction; (d), (e), (f) gate output corresponding to the experts for notes C, A#, and A, respectively.

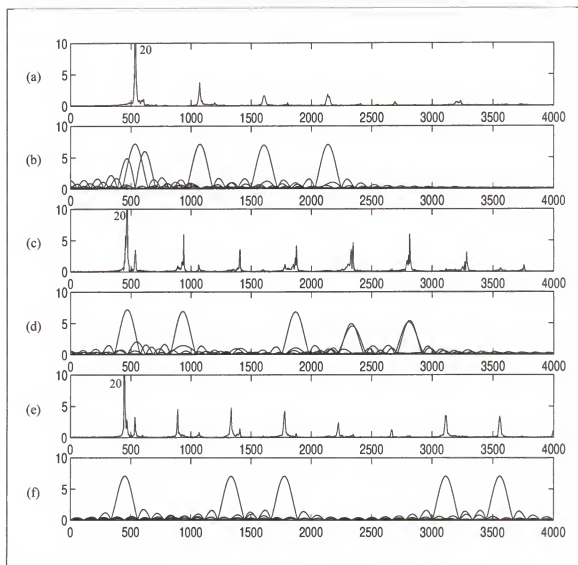


Figure 5-10. Violin: (a),(c),(e) original signal periodogram (linear) for notes C, A#, and A, respectively, vs. frequency in Hz; (b),(d),(f) corresponding expert eigenfilter frequency response (linear) vs. frequency (Hz).

5.3 Temporal PCA

In order to fully understand the simulation results of competitive temporal PCA, we must understand temporal PCA. This can be best accomplished by understanding the frequency domain behavior of the eigenfilters. The discrete time Fourier transform (DTFT) of the eigenvalue equation results in an integral equation

$$Ru = \lambda u \Leftrightarrow \int_{-1/2}^{1/2} \frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} S(f') \psi(f') df' = \lambda \psi(f) \quad (5.2)$$

where $S(f)$ is the power spectrum corresponding to the autocorrelation function, $r(n)$,

$$r(n) = \int_{-1/2}^{1/2} S(f) e^{i2\pi n f} df. \quad (5.3)$$

The eigenfunction $\psi(f)$ is the DTFT of the eigenvector u

$$\psi(f) = \varepsilon \sum_{n=0}^{N-1} u(n) e^{i2\pi f[n-(N-1)/2]}. \quad (5.4)$$

Since u is either symmetric or anti-symmetric, the phase term $(N-1)/2$ and the constant ε are introduced to ensure that the eigenfunction is real. The constant ε is defined to be 1 or i , depending on whether u is symmetric or anti-symmetric, respectively.

Equation (5.2) in the frequency domain is a homogenous Fredholm equation of the second kind with a Dirichlet kernel and weighting function $S(f)$. When $S(f)$ is a bandlimited low pass filter, such that $S(f)=1$ for $|f| \leq W$, and 0 elsewhere, the resulting equation defines the discrete prolate spheroidal wave functions (DPSWF) of Slepian (1978)

$$\int_{-W}^W \frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} U_k(f', N, W) df' = \Lambda_k(N, W) U_k(f, N, W) \quad k = 0 \dots N-1. \quad (5.5)$$

Properties of the DPSWF, $U_k(f, N, W)$, and their eigenvalues, $\Lambda_k(N, W)$, are given in the appendix. We will now propose and demonstrate a close relationship between the frequency domain solutions of (5.2) and the DPSWF for the medium window case. The large and small window solutions of (5.2) have been studied from various perspectives, which we briefly summarize below.

5.3.1 The Large Window Solution

As $N \rightarrow \infty$, the Dirichlet function becomes a Dirac delta function and (5.2) becomes

$$\lambda \psi(f) = \psi(f) S(f) \quad (5.6)$$

the solution of which is

$$\psi_k(f) = \delta(f - f_k), \lambda_k = S(f_k). \quad (5.7)$$

That is, the eigenfilters form a fixed Fourier basis, and the eigenvalues become the power spectrum. Since the autocorrelation matrix is real and symmetric, the eigenvalues are real, and we can enforce the constraint that the eigenfilters be real. Then, each eigenvalue will be doubly degenerate, associated with two real sinusoidal eigenvectors separated in phase by 90 degrees. Assuming that the eigenvalues have been ordered from largest to smallest, so that λ_0 is the largest eigenvalue, then λ_0 is the maximum value of the power spectrum: $\lambda_0 = \max_f [S(f)]$.

5.3.2 The Small Window Solution

Casdagli et al. (1991) showed that when the autocorrelation matrix is viewed as a sampled version of the continuous time autocorrelation function, the time domain eigenvectors can be approximated by the discrete Legendre polynomials when the window width, $N \tau_s$, is small, where τ_s is the sampling period. For the purely discrete case, as N becomes small, the time-bandwidth product law dictates that the bandwidth of the eigenfilters will become

large. The eigenfilters then form a fixed basis in the sense that they become increasingly insensitive to the specific spectral properties of the signal. It is interesting to note that in the limit as the bandwidth approaches the extremes $W \rightarrow 0$ and $W \rightarrow \frac{1}{2}$, the DPSWF, become the Legendre polynomials.

5.4 Motivation: Matched Filter Approach

Thomson (1982) first suggested that the DPSWF might be a suitable basis in which to expand the eigenfunctions in the frequency domain. We now consider an energy maximization approach to solving the eigenvalue equation in the frequency domain which presents a strong argument for the use of the DPSWF. It is well known that maximizing or minimizing

$$E = \mathbf{u}^T \mathbf{R} \mathbf{u} = \int_{-1/2}^{1/2} |\psi(f)|^2 S(f) df \quad (5.8)$$

subject to the constraint

$$\mathbf{u}^T \mathbf{u} = \int_{-1/2}^{1/2} |\psi(f)|^2 df = 1 \quad (5.9)$$

leads to the eigenvalue equation (5.2). Equation (5.8) in the frequency domain shows that the eigenfunctions act as matched filters. The frequency domain solution to maximizing E subject to (5.9) is the same as the infinitely large window solution; namely, a Dirac delta function, $\psi(f) = \delta(f - f_{\max})$, centered around the peak value of the signal spectrum, $f_{\max} = \max_f [S(f)]$, so that E simply picks out the largest value of $S(f)$. However, since \mathbf{u} is index limited in the time domain, it cannot have infinite concentration in the frequency domain. This means that the constraint that \mathbf{u} is index limited is *not* embodied in the fre-

quency domain eigenvalue equation. The Dirichlet kernel arises because of the truncation of the infinite sequence $r(n)$ but the filter u is *not* the truncation of a longer sequence. Still, this suggests that the solution must be related to the index limited sequence that has the largest concentration of energy in the frequency domain. This is, in fact, a defining property of the DPSWF with the largest eigenvalue. Therefore, we propose that the solution to the maximization of (5.8) is, to a high degree of approximation, a frequency shifted DPSWF

$$E_{max} \approx \max_{f_0, W_0} \int_{-1/2}^{1/2} |U_0(f - \hat{f}_0, N, W_0)|^2 S(f) df. \quad (5.10)$$

Denote the optimal values by \hat{f}_0 and \hat{W}_0 . Since the DPSWF's are real, this implies that the solution to (5.2) corresponding to the largest eigenvalue is approximately

$$\psi_0(f) \approx U_0(f - \hat{f}_0, N, \hat{W}_0). \quad (5.11)$$

Since both $\psi_0(f)$ and $U_0(f)$ have all their zeros on the unit circle (Makhoul, 1981), the frequency shifting must be done so as to preserve this property. Also, note that since U_0 is symmetric, if the spectrum is also symmetric about its local maximums, then \hat{f}_0 will coincide with one of these local maximums.

We now demonstrate some of these properties for a spectrum from the switching FIR simulation of the first part of this chapter. Figure 5-11 (a) shows the original spectrum. Panel (b) shows the spectrum of the first eigenfilter and panel (c) shows the spectrum of the optimally shifted DPSWF. We see that there is a remarkable similarity in the two spectra. This is also confirmed in the time domain, shown in Figure 5-12. Again, there is a remarkable similarity between the 1st eigenvector and the optimally frequency shifted DPSS.

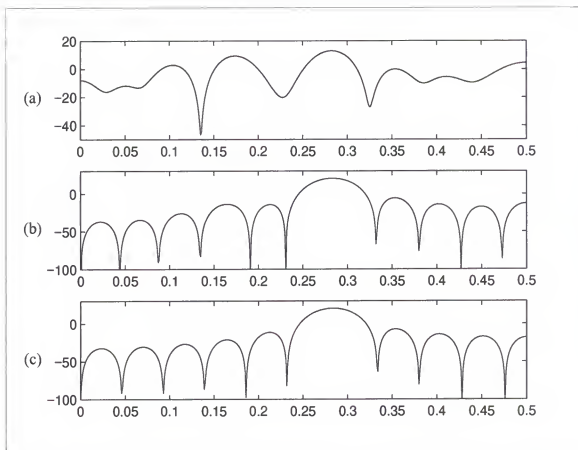


Figure 5-11. Power spectra vs. frequency (digital): (a) original signal; (b) 1st eigenfilter; (c) optimal frequency shifted DPSWF.

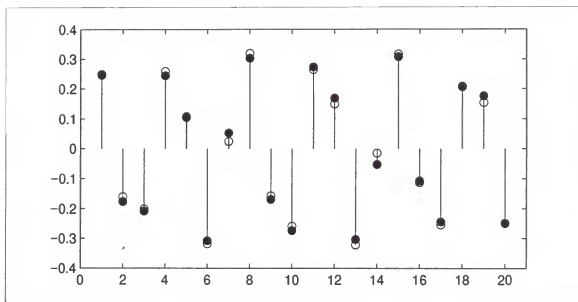


Figure 5-12. Signal 1st eigenfilter (solid) and optimal frequency shifted DPSWF (hollow) vs. time (samples).

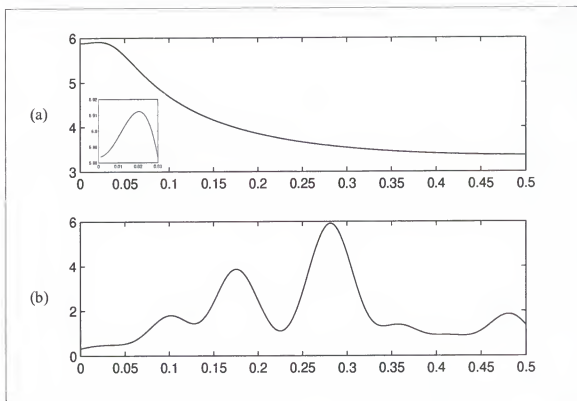


Figure 5-13. (a) Matched energy vs. W for optimal frequency offset \hat{f}_0 ; (b) Matched energy vs. \hat{f}_0 for optimal half-bandwidth \hat{W} .

Figure 5-13 shows the matched energy as a function of the frequency offset (a) and half-bandwidth parameter (b) of the DPSWF. Note that the optimal DPSWF is relatively insensitive to the half-bandwidth parameter, W , and peaks at a value close to the optimal time-bandwidth product, $2WN \approx 2 \cdot 0.02 \cdot 20 = 0.8$. This implies that the value of W can be chosen based on the dimensionality, N , of the tapped delay line, irrespective of the particular signal spectrum. Note also that the optimal matched energy is very sensitive to the frequency offset, a desirable property.

5.5 Separable Kernel Approach

The previous development presented a strong motivation for considering the DPSWF as a suitable basis for the eigenfunctions. We now proceed to find such a basis. A closed

form algebraic solution of a homogenous Fredholm equation of the second kind can be found whenever the kernel is separable or degenerate, meaning expressible as a sum of products of the form

$$\frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} = \sum_{n=0}^{N-1} Q_n(f)Q_n(f'). \quad (5.12)$$

Substitution back into the frequency domain eigenvalue equation leads to

$$\sum_{n=0}^{N-1} Q_n(f) \int_{-1/2}^{1/2} Q_n(f')S(f')\psi(f')df' \equiv \sum_{n=0}^{N-1} c_n Q_n(f) = \lambda \psi(f) \quad (5.13)$$

where the integral has been reduced to the constant c_n . Note that (5.13) represents an expansion of the eigenfunctions in terms of the basis $Q_n(f)$

$$\psi(f) = \frac{1}{\lambda} \sum_{n=0}^{N-1} c_n Q_n(f). \quad (5.14)$$

We can solve for the expansion coefficients by multiplying both sides of (5.13) by $Q_m(f) \cdot S(f)$ and integrating:

$$\sum_{n=0}^{N-1} c_n \int_{-1/2}^{1/2} Q_m(f)Q_n(f)S(f)df \equiv \sum_{n=0}^{N-1} a_{mn}c_n = \lambda c_m. \quad (5.15)$$

Regarding the expansion coefficients, $\{c_n\}$, of the eigenfunction as a vector, they are themselves the solution of an eigenvalue equation of a matrix A

$$Ac = \lambda c \quad (5.16)$$

whose elements are given by

$$a_{mn} = \int_{-1/2}^{1/2} Q_m(f) Q_n(f) S(f) df. \quad (5.17)$$

There will be N solutions to (5.16), each producing a different solution to (5.2). There are two known ways to separate the Dirichlet kernel. One is through the use of a Fourier basis

$$\frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} = \sum_{n=0}^{N-1} e^{i\pi(N-1-2n)f} e^{-i\pi(N-1-2n)f'}. \quad (5.18)$$

Use of (5.18) in conjunction with the previous development leads to $\mathcal{A} \Rightarrow \mathcal{R}$ and thus back to the *time domain* eigenvalue equation (5.2) and thus yields no new insight. However, the Dirichlet kernel is also separable in terms of the DPSWF

$$\frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} = \sum_{n=0}^{N-1} U_n(f-f_0, N, W) U_n(f'-f_0, N, W). \quad (5.19)$$

where f_0 is an arbitrary frequency offset. Use of (5.19) in conjunction with the previous development leads naturally to an expansion of the k^{th} eigenfunction in terms of the DPSWF,

$$\Psi_k(f) = \frac{1}{\lambda_k} \sum_{n=0}^{N-1} (c_k)_n U_n(f-f_0, N, W) \quad k = 0 \dots N-1 \quad (5.20)$$

where $(c_k)_n$ is the n^{th} component of the k^{th} vector solution of the eigenvalue equation, $\mathcal{A} \mathbf{c}_k = \lambda_k \mathbf{c}_k$, with \mathcal{A} having elements

$$a_{mn} = \int_{-1/2}^{1/2} U_m(f-f_0, N, W) U_n(f-f_0, N, W) S(f) df. \quad (5.21)$$

Thus, we see that an expansion of the eigenfunctions of (5.2) in terms of the DPSWF of (5.5) falls out naturally as a result of the separability of the Dirichlet kernel. We now turn to the question of choosing the frequency offset. Based on the matched filter approach, it seems natural to choose f_0 to coincide with the local peak of the signal spectrum with the largest energy, and W to be the effective bandwidth of the peak. This will ensure that energy is concentrated along the main diagonal of A , and that the principal eigenfunction will be expressed primarily in terms of the principal DPSWF, $\psi_0(f) \approx U_0(f - \hat{f}_0, N, \hat{W})$, and that the other expansion coefficients will be nearly zero.

5.6 Expansion in Terms of Multiple Frequency Shifted DPSWF

We now present a generalization of Thomson's proposed DPSWF basis. We start with (5.13), an expansion of the eigenfunctions in terms of a basis

$$\psi(f) = \sum_{n=0}^{N-1} c_n Q_n(f) \quad (5.22)$$

where the basis is a set of frequency shifted DPSWF. We divide the digital frequencies into K frequency bands, each with a center frequency of f_k . Each band may have a different bandwidth, specified by the half-bandwidth parameter W_k . For each center frequency, we use only the first $2NW_k$ frequency shifted DPSWF, since only those have eigenvalues close to one. The basis is then given by:

$$Q_m(f) = U_l(f - f_k, N, W_k) \quad \begin{array}{l} k = 0 \dots K-1 \\ l = 0 \dots (2NW_k - 1) \end{array} \quad (5.23)$$

The index m is the cumulative count (minus one) of the number of DPSWF employed, of which the total number is

$$\sum_{k=0}^{K-1} 2NW_k = N \sum_{k=0}^{K-1} 2W_k = N. \quad (5.24)$$

as required in order that the number of equations equal the number of eigenfunctions. If the quantity $2NW_k$ is not an integer, it can be rounded up or down as required, as long as the total number of basis functions is N . This division of the digital frequency axis is shown in Figure 5-14. (a). Note that since the power spectrum is an even function, we must have $f_k = f_{K-1-k}$ and $W_k = W_{K-1-k}$. Thomson's proposed basis is a subset of this where all the bandwidths are equal, and is shown in Figure 5-14. (b).

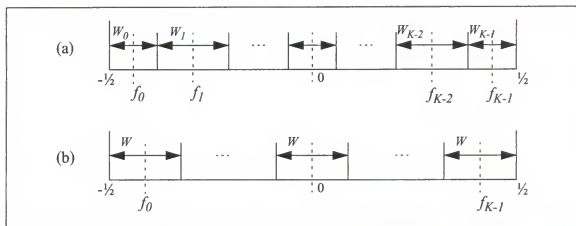


Figure 5-14. Dividing the digital frequency axis: (a) arbitrary bandwidth. (b) Thomson's uniform bandwidth.

It is important to note that this basis, while orthogonal within bands, is not orthogonal between bands. However, as long as the basis functions are linearly independent, they will be complete in the space of Fourier transforms of index limited sequences. The linear independence of the basis functions can theoretically be tested by verifying that the Wronskian is non-zero

$$\det \begin{bmatrix} Q_0(f) & \cdots & Q_{N-1}(f) \\ Q'_0(f) & \cdots & Q'_{N-1}(f) \\ \vdots & \cdots & \vdots \\ Q_0^{(N-1)}(f) & \cdots & Q_{N-1}^{(N-1)}(f) \end{bmatrix} \neq 0. \quad (5.25)$$

Unfortunately, since there is no analytical expression for the DPSWF, it is impossible to evaluate the Wronskian for the general case.

We now proceed to solve for the expansion coefficients. Substituting (5.22) into the frequency domain eigenvalue equation (5.2) there results

$$\sum_{n=0}^{N-1} c_n \int_{-1/2}^{1/2} \frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} S(f') Q_n(f') df' = \lambda \sum_{k=0}^{N-1} c_k Q_k(f). \quad (5.26)$$

Multiplying both sides by $Q_n(f)$ and integrating over the principal domain of the shifted DPSWF, the left hand side of (5.25) becomes

$$\sum_{m=0}^{N-1} c_m \int_{-1/2}^{1/2} \left\{ \int_{f_k - W_k}^{f_k + W_k} \frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} Q_n(f) df \right\} S(f') Q_m(f') df'. \quad (5.27)$$

The integral in brackets can be reduced by using the defining equation for the DPSWF (5.5), in which case (5.26) becomes

$$\sigma_{m'} \sum_{m=0}^{N-1} c_m \int_{-1/2}^{1/2} Q_{m'}(f') Q_m(f') S(f') df' = \lambda \sum_{m=0}^{N-1} c_m \int_{f_k - W_k}^{f_k + W_k} Q_{m'}(f) Q_m(f) df \quad (5.28)$$

where it is to be understood that the eigenvalue associated with the m^{th} basis function is a DPSWF eigenvalue that depends on both the particular frequency band index, k , and the DPSWF index l within the band: $\sigma_m = \Lambda_l(N, W_k)$. We can write (5.28) as

$$\sigma_{m'} \sum_{m=0}^{N-1} a_{m'm} c_m = \lambda \sum_{m=0}^{N-1} b_{m'm} c_m \quad (5.29)$$

or in matrix form, as a generalized eigenvalue equation

$$\Sigma A c = \lambda B c \quad (5.30)$$

where c is the vector of expansion coefficients of the eigenfunction, Σ is a diagonal matrix containing the DPSWF eigenvalues, and A and B are matrices with elements, respectively,

$$a_{m'm} = \int_{-1/2}^{1/2} Q_{m'}(f) Q_m(f) S(f) df \quad (5.31)$$

$$b_{m'm} = \int_{f_k - W_k}^{f_k + W_k} Q_{m'}(f) Q_m(f) df. \quad (5.32)$$

In general, there will be N solutions to (5.30), each producing a unique solution to (5.2).

Our goal now is to specify a basis for which the eigenfunctions are expressible in terms of the minimum number of DPSWF. That is, we desire that each vector of expansion coefficients, c_n , be dominated by a single coefficient. We saw from the matched filter approach that the eigenfunction with the largest eigenvalue was approximately expressible in terms of a single DPSWF when the DPSWF was frequency shifted to coincide with the local peak of the signal spectrum with the largest energy. Based on that result, we propose a basis of frequency shifted DPSWF where the K center frequencies and bandwidths are chosen to align with the local peaks of the signal spectrum. This should concentrate most of the energy of the expansion in as few coefficients as possible, and result in the closest possible relationship between the eigenfunctions and the DPSWF.

5.7 Conclusion

We've seen how the Competitive PCA algorithm from the previous chapter can be adapted to time series by simply adding a tapped delay line at the input. The algorithm was then able to segment time series, but not as well as the classical GLR algorithm performing prediction. Therefore, we conclude that its true strength lies in its potential for noise reduction and optimal encoding.

In order to understand the simulation results, we then explored the properties of an eigenfilter in isolation. To this end, we examined the eigenvalue equation in both the time and frequency domain. We showed that solving the eigenvalue equation in the time domain is equivalent to assuming a Fourier basis in the frequency domain. We also showed that there is another natural basis for solving the equations; namely, the discrete prolate spheroidal wave functions (DPSWF). We then proposed conditions under which there is a close correspondence between the eigenfilters and the DPSWF.

CHAPTER 6

MEMORY IN GATED COMPETITIVE SYSTEMS

6.1 Introduction

We've seen how the output of the gate effectively segments a signal. Unfortunately, due to several factors, the segmentation can exhibit spurious switching. Such switching may be due to genuine overlap in the return maps of the various dynamical regimes or, in the language of non-linear dynamics, the manifolds of the various dynamical regimes may intersect in phase space. Additionally, measurement noise can "smear" manifolds that ordinarily would never intersect.

One way to reduce such spurious switching is to add memory to our gated competitive system. Through the use of memory, an expert that has performed well in the recent past can be given an advantage in the present. In general, memory can be added either to the competitive framework (gate), the competitive mechanism (the experts' performance measure), or the experts themselves. We have already seen one architecture that adds memory to the gate; namely, the Annealed Competition of Experts (ACE). In this chapter we will consider other ways to add memory to a gated competitive system.

6.2 Adding Memory to a Cost Function

We now present a recursive mean square error cost function, and show that it leads to momentum learning. The standard mean square error criteria for a single adaptive system is a summation of the squared errors over the entire data set:

$$C = \sum_{n=1}^N e^2(n) \quad (6.1)$$

where the instantaneous error $e(n)$ is the difference between the system output and the desired output. Now consider a cost function that uses a recursive estimate of the local mean square error

$$C = \sum_{n=1}^N \varepsilon(n) \quad (6.2)$$

where $\varepsilon(n)$ is the local mean square error calculated using a recursive filter

$$\varepsilon(n) = \lambda e^2(n) + (1 - \lambda)\varepsilon(n-1) \quad 0 < \lambda < 1. \quad (6.3)$$

The parameter λ controls the memory depth of the filter. There are several ways to define the effective memory of a recursive system, but the simplest is to calculate the average of the temporal index weighted by the impulse response (Principe et al., 1993). According to this definition, it is easy to show that the effective memory depth of (6.3), in samples, is given by λ^{-1} .

Taking the expected value of both sides of (6.3) and simplifying shows that it is an unbiased estimate of the mean square error: $E[\varepsilon] = E[e^2]$. Now consider the gradient of the cost function with respect to an adaptable weight

$$\frac{\partial C}{\partial w} = \sum_{n=1}^N \frac{\partial}{\partial w} \varepsilon(n) \quad (6.4)$$

where the instantaneous gradient is given by

$$\frac{\partial}{\partial w} \varepsilon(n) = 2\lambda \frac{\partial}{\partial w} e(n) + (1 - \lambda) \frac{\partial}{\partial w} \varepsilon(n-1). \quad (6.5)$$

For an on-line adaptive system, the weight change is proportional to the instantaneous gradient. Viewed in this way, (6.5) is equivalent to *learning with momentum*. In the neural network literature, momentum learning is presented as an ad-hoc way of speeding up learning. Here, we see it is the natural consequence of a recursive mean square error cost function, which we will now employ in the following models.

6.3 Neighborhood Map of Competing Predictors

Our implementation is shown in Figure 6-1. A set of predictors works operate in parallel on the input time series. The winning predictor is the one with the smallest local mean squared error

$$winner(t) = \operatorname{argmin}_i[\varepsilon_i(t)] \quad (6.6)$$

where the local mean square error is computed using the recursive estimate of (6.3) for each expert

$$\varepsilon_i(n) = \lambda e_i^2(n) + (1 - \lambda)\varepsilon_i(n - 1) \quad 0 < \lambda < 1 \quad (6.7)$$

where e_i is the instantaneous squared error of the i^{th} predictor. The memory term, λ , is identical for all experts, but can be manually adjusted during training. The gating function, which moderates the learning rate of the predictors, is determined by the distance from the winning predictor to the other predictors

$$g_i(t) = e^{\left(\frac{d_{i,j}^2(t)}{2\sigma^2(t)}\right)} \quad (6.8)$$

where i is the predictor to be updated, j is the winning predictor, $d_{i,j}$ is the neighborhood distance from predictor i to predictor j , and σ is an annealing parameter which controls the

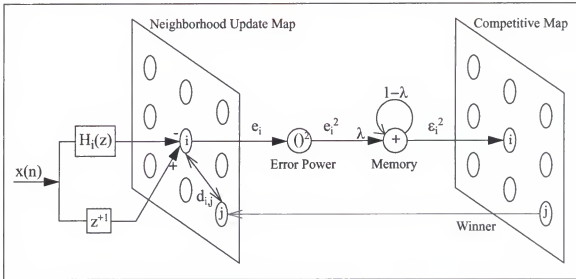


Figure 6-1. Neighborhood map of competing predictors.

neighborhood width. This results in the model shown in Figure 6-1. The signal flow graph is shown for only one predictor.

In exact analogy with training a Kohonen map, both the neighborhood width and the global learning rate are annealed during training according to an exponentially decreasing schedule

$$\sigma(t) = \sigma_0 e^{-\frac{t}{\tau}} \quad \mu(t) = \mu_0 e^{-\frac{t}{\tau}} \quad (6.9)$$

where τ is the annealing rate. The overall learning rate for the i^{th} predictor is thus given by

$$\eta_i(t) = \mu(t) \cdot g_i(t). \quad (6.10)$$

6.3.1 Simulation I: Switching FIR Process

In order to better understand the algorithm, we used linear predictors trained with the normalized LMS algorithm for the following simulation. For viewing convenience, we used a one dimensional continuous map, where the last node is mapped as a neighbor to the first node. Training was conducted for 50 passes through the entire time series, using an annealing rate of $\tau = 5$ passes. The time series was a switching FIR process. It was gen-

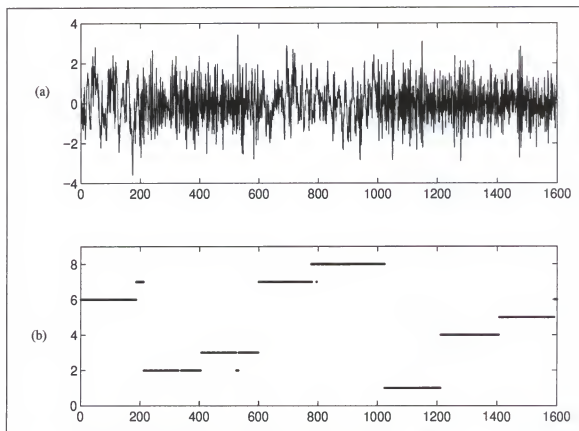


Figure 6-2. Switching FIR process: (a) time series; (b) winning predictors after training. erated by a 6th order normalized FIR filter, driven by zero mean Gaussian noise of unit variance, whose coefficients were random and change every 200 samples. The time series, which consists of a total of 8 stationary regions, is shown in Figure 6-2.

For the predictors, we used eight 8th order predictors. The memory depth of the error was 25 samples. The initial neighborhood and learning rate were 16 and 1, respectively. The winning predictors after training are shown in Figure 6-2 (b). The time series was fairly successfully segmented, except for a longer than expected winning run for predictor 8, and minor switching errors near samples 200 and 475. Those latter errors correlate well with outliers in the time series, which temporarily degrades the performance of the winning predictor. However, in both cases, recovery occurred rapidly.

From the FIR coefficients that produced the series, we can define a similarity coefficient, which is tabulated in Table 6-1.

$$|\cos \theta_{i,j}| = |W_i \cdot W_j|. \quad (6.11)$$

Table 6-1. Distance between the eight stationary regions.

Region	I	II	III	IV	V	VI	VII	VIII
I	1							
II	.38	1						
III	.36	.61	1					
IV	.86	.51	.40	1				
V	.75	.23	.17	.79	1			
VI	.07	.38	.03	.40	.59	1		
VII	.25	.46	.63	.25	.38	.42	1	
VIII	.65	.59	.69	.43	.39	.01	.57	1

Region I refers to samples 1-200, region II refers to samples 201-400, and so on. Using this metric, the two regions with the highest similarity (0.86) were sample regions 1-200 and 601-800. From the neighborhood map, we indeed find that neighboring predictors 6 and 7 won those regions. Likewise, the two regions with the second highest similarity (0.79) were sample regions 601-800 and 801-1000, won by neighboring predictors 7 and 8. Alternatively, we find that the two regions the most dissimilar (0.01) were sample regions 1001-1200 and 1401-1600. Indeed, the winners of those two regions are predictors 1 and 5, as far apart as the map will allow. Overall, there is a very high correlation between the similarity of two regions and their distance on the neighborhood map.

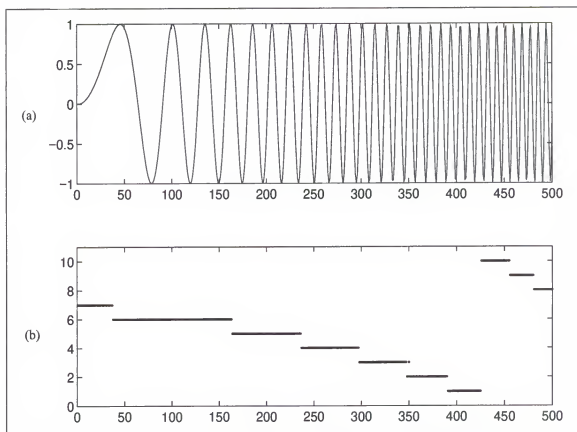


Figure 6-3. Chirp example: (a) time series; (b) winning predictors after training.

6.3.2 Simulation II: A Non-stationary Time Series

For this simulation the time series consisted of a chirp signal of 500 points, whose angular frequency varied from 0 to $1/8$ of the Nyquist frequency. The signal is shown in Figure 6-3 (a). For the predictors, we used ten 6th order predictors. The memory depth of the error was 2.5 samples. The initial neighborhood and learning rate were 10 and 1, respectively. The winning predictors after training are shown in Figure 6-3 (b).

We see that the non-stationary time series was segmented in an intuitive way. Except for the first winning predictor, the number of samples per segmented region decreases with time. Also, similar frequency ranges in the time series are mapped as neighbors in the predictor map.

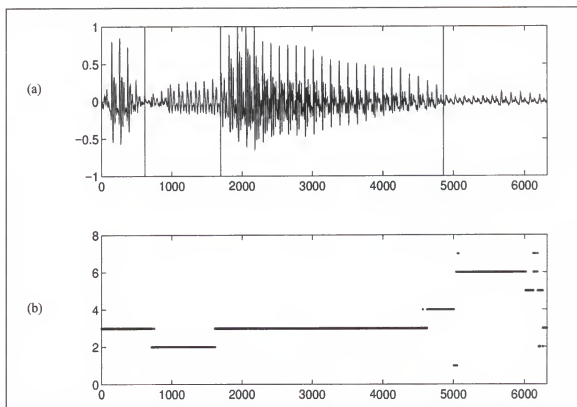


Figure 6-4. Speech example: (a) time series and phoneme segmentation; (b) winning predictors.

6.3.3 Simulation III: Phoneme Segmentation of Speech

For this simulation the time series consisted of the word “alone” spoken by a male speaker and sampled at 16 kHz. The signal, which was extracted from the Timit database, is shown along with its suggested phoneme segmentation (a/l/ow/n) in Figure 6-4.

For the predictors, we used seven 12th order predictors, in order to allow for transitions between the four phonemes. The memory depth of the error was 200 samples. The initial neighborhood and learning rate were 128 and 1, respectively. The winning predictors after training are shown in Figure 6-4 (b).

The segmentation corresponds fairly closely to the Timit suggested phoneme segmentation, allowing for a transition region between the third and fourth phonemes. However, although properly segmented, the same predictor won both the /a/ and /ow/ phonemes.

This simulation also illustrates how predictors can drop out of the competition, as predictors one and seven win less than 1% of the time, so that only five of the original seven predictors are finally active.

6.4 Self-annealing Competitive Prediction

Recall that the ACE algorithm uses an output based gate with memory. The memory is in the form of a boxcar moving average (MA) of the local squared error of the experts, to which a Gaussian function is applied. The variance of the Gaussian function determines the degree of competition. In the ACE algorithm, memory is used in the gating function only. Here we propose a cost function that imbeds memory in both the gate and the experts' mean square error

$$c(n) = \sum_{i=1}^K g_i(n) \varepsilon_i(n) \quad (6.12)$$

where ε_i is the local recursive mean square error of the i^{th} predictor, repeated here for convenience

$$\varepsilon_i(n) = \lambda e_i^2(n) + (1 - \lambda) \varepsilon_i(n-1) \quad 0 < \lambda < 1. \quad (6.13)$$

Again, $e_i(n)$ is the instantaneous error of the i^{th} expert. All the experts use the same value for the memory parameter, λ , but the algorithm could be modified to allow for differing memories.

6.4.1 Coupling the Competition with the Memory Depth

Turning to the gate, in the ACE algorithm, the degree of competition within the gate, as expressed through the annealing parameter, and the memory depth of the squared error are uncoupled. And yet, it is intuitive that the longer the time period over which we collect

information about the performance of the experts, the better should be our judgement about which expert is valid. Such a coupling eliminates the need for separate annealing of the memory depth and the competition. Niedzwiecki (1992) has in fact formulated such a relationship. Assuming that the variances of the errors of the experts are unknown but distributed according to a so-called non-informative prior distribution, Niedzwiecki's formula for the probability of the i^{th} expert is given by

$$g_i(n) = \frac{\varphi_i(n)}{\sum_{j=1}^K \varphi_j(n)} \quad (6.14)$$

where the unnormalized gate output is given by

$$\varphi_i(n) = [\varepsilon_i(n)]^{\frac{-M}{2}} \quad (6.15)$$

and ε_i is the causal boxcar MA of the squared error

$$\varepsilon_i(n) = \frac{1}{M} \sum_{m=0}^{M-1} e_i^2(n-m). \quad (6.16)$$

The unnormalized gate given by equation (6.15) embodies the coupling between memory depth and degree of competition. When $M=1$, the unnormalized gate is just the absolute value of the instantaneous error $\varphi_i(n) = |e_i(n)|$. As $M \rightarrow \infty$, the gate implements hard competition such that $g_i(n)=1$ when i represents the expert with the smallest mean square error, and $g_i(n)=0$ for all others.

Niedzwiecki developed his formulas for evaluating the performance of *known* experts. Here, we borrow Niedzwiecki's results and incorporate it into our cost function (6.12) for adaptable experts. We use (6.14) as the gating function, except that we replace the moving

average calculation of the mean square average in (6.16) by our recursive calculation in (6.13). Likewise, we replace the memory depth, M , in (6.15) by the memory parameter, λ^{-1} ,

$$\varphi_i(n) = [\varepsilon_i(n)]^{\frac{-1}{2\lambda}}. \quad (6.17)$$

6.4.2 Gradient Descent

We then do gradient descent on *both* the calculated gate and the mean square errors of (6.12). Taking the partial derivative with respect to any free parameter, w_k , in the k^{th} system, results in:

$$\Delta w_k = -\eta \frac{\partial}{\partial w_k} E(n) = -\eta g_k \left[\frac{E(n)}{\varepsilon_k(n)} + 2\lambda - 1 \right] z_k(n) \quad (6.18)$$

$$z_k(n) = e_k(n) \frac{\partial}{\partial w_k} y_k(n) + (1 - \lambda) z_k(n-1) \quad (6.19)$$

and with respect to the λ parameter

$$\Delta \lambda = \frac{-\eta}{2\lambda} \sum_{k=1}^K g_k \left[\left(\frac{E}{\varepsilon_k} + 2\lambda - 1 \right) v_k(n) - \frac{[E - \varepsilon_k] \log \varepsilon_k(n)}{\lambda} \right] \quad (6.20)$$

$$v_k(t) = \frac{\varepsilon_k(n) - \varepsilon_k(n-1)}{\lambda} + (1 - \lambda) v_k(n-1). \quad (6.21)$$

Equation (6.19) for z_k is the standard weight update equation with momentum for the k^{th} expert operating independently. Thus, momentum learning, which has been previously presented as an ad-hoc way to speed up and stabilize neural network training, falls out naturally as a result of using the mean square error in the cost function (6.12) instead of the instantaneous error.

The competitive nature of the algorithm is evident in (6.18), where the total weight update is given by the product of z_k with the probability of the k^{th} expert, g_k , and another term which depends on the inverse of the mean square error of the k^{th} expert. Thus, the expert with the smallest mean square error will have the largest weight update. Furthermore, (6.18) also shows how the annealing is coupled with the memory depth. For $\lambda > 0.5$, the term in brackets is always positive, and thus all the experts move towards the data to improve their predictions. However, For $\lambda < 0.5$, the sign of the term in brackets can be either positive or negative, depending on whether the mean square error of the k^{th} predictor is less or greater than, respectively, the total cost. Thus, experts that perform poorly can actually be pushed away from the data, although at a small learning rate due to the gating function.

6.4.3 Simulation

To test the algorithm, we used the same switching FIR process as in the previous simulation. We also used the same number and type of predictors. The initial value of λ was 1, and after 30 iterations, it had evolved to a value of 0.6, corresponding to a memory depth of approximately 1.6 samples. The winning predictors after training are shown in Figure 6-5 (b).

The time series was partially segmented, except there is more spurious switching than there was for the Neighborhood Map of Predictors. This is because λ did not adapt to a small enough value to invoke hard competition, whereas in the previous algorithm, it was experimentally set to a value which gave a stable segmentation. Still, the results are encouraging.

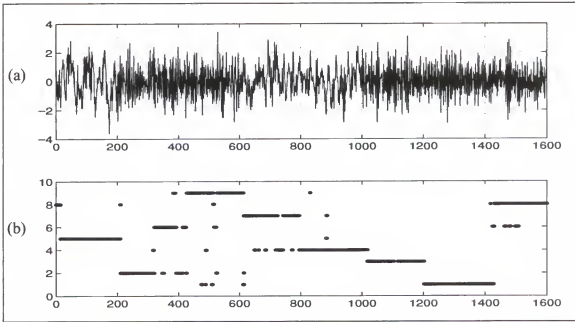


Figure 6-5. Switching FIR process: (a) time series; (b) winning predictors after training.

6.5 Adding Static Memory to the Mixture of Experts

We now seek to add memory to the experts' pdf's. We will do this for the Mixture of Experts model, which was explained in Chapter 3. The key point in advancing this theory is to regard the performance of the experts over some time window, v , as a *single event* characterized by their local root mean square error. Furthermore, the gate sees all the data that was presented to the experts over this time window, and produces a single output, representing the apriori probabilities of the various experts for this single event. In this way, the gate makes its decision based on all the data that the experts have processed.

This is fundamentally different from viewing a set of expert predictions over some time window as an ensemble. There, the total probability of the ensemble is the product of the individual probabilities, as calculated through the likelihood product of Gaussian probabilities. Here, we determine the probability as calculated by the root mean square error performance based on a chi-square distribution.

6.5.1 Adding Static Memory to the Experts' pdf

The performance of each expert will be modeled by its local root mean square error over time. If the instantaneous error is Gaussian distributed, then the root mean square error is chi-square distributed. The local root mean square error of the i^{th} expert is

$$\chi_k(t) = \sqrt{\varepsilon_k(t)} \quad (6.22)$$

$$\varepsilon_k(t) = \frac{1}{v} \sum_{\tau=0}^{v-1} e_k^2(t-\tau) \quad (6.23)$$

where e_k is the instantaneous error of the k^{th} expert, and v is the memory depth.

6.5.2 Chi-Square Distribution

Dropping reference to the particular expert momentarily, if e is Gaussian distributed, then χ is distributed according to the chi-square distribution

$$p(\chi) = \frac{1}{K} \chi^{\left(\frac{v}{2}-\frac{1}{2}\right)} e^{-\frac{v\chi}{2\sigma^2}} \quad (6.24)$$

where K is a normalizing constant given by

$$K = \frac{1}{2} \left(\frac{2\sigma^2}{v} \right)^{\frac{v}{2}} \text{Gamma} \left[\frac{v}{2} \right]. \quad (6.25)$$

The expected value of the of the mean square error is the Gaussian distribution's variance

$$E[\chi^2] = E[\varepsilon] = \sigma^2. \quad (6.26)$$

6.5.3 Gradient Descent

The likelihood is now formed from windows of data of size v , in order to ensure statistical independence

$$C = -\ln L = \sum_{n=v, 2v \dots}^N -\ln \sum_{k=1}^K h_k(n) p(d(n)|x(n), k) . \quad (6.27)$$

We perform gradient descent on (6.27), except now the expert's pdf's are chi-square distributions. First, we need the partial derivative of the expert's chi-square distributions with respect to some free parameter, w_k , in the k^{th} expert

$$\frac{\partial p_k}{\partial w_k} = \left[\frac{1 - \frac{1}{v}}{\varepsilon_k} - \frac{1}{\sigma_k^2} \right] p_k \sum_{\tau=0}^{v-1} e_k(t-\tau) \frac{\partial}{\partial w_k} y_k(t-\tau) . \quad (6.28)$$

We then plug (6.28) into (3.20) to obtain

$$\frac{\partial C}{\partial w_k} = \sum_{t=v, 2v \dots}^N h_k(t) \left[\frac{1}{\sigma_k^2} - \frac{1 - \frac{1}{v}}{\varepsilon_k(t)} \right] \left[\sum_{\tau=0}^{v-1} e_k(t-\tau) \frac{\partial}{\partial w_k} y_k(t-\tau) \right] . \quad (6.29)$$

Note that the sum within the right most brackets represents the total weight change over the time window for an expert operating independently. Also, the reason for modeling the root mean square error (as opposed to the mean square error) becomes clear once it is seen that (6.29) reduces to the equivalent Mixture of Experts equation (3.20) when $v=1$.

There is still an analytical solution for the variances, which is now expressed in terms of the expert's local mean square error

$$\frac{\partial C}{\partial \sigma_k^2} = 0 \Rightarrow \sigma_k^2 = \frac{\sum_{t=v, 2v \dots}^N h_k(t) \varepsilon_k(t)}{\sum_{t=v, 2v \dots}^N h_k(t)} . \quad (6.30)$$

Based on (6.26), the winning expert will be the one whose local mean square error best matches its variance, in which case the algorithm reduces to a block form of the ME algorithm

$$\varepsilon_k \approx \sigma_k^2 \Rightarrow \frac{\partial C}{\partial w_k} \approx \sum_{t=v, 2v, \dots}^N \frac{h_k(t)1}{\sigma_k^2 v} \left[\sum_{\tau=0}^{v-1} e_k(t-\tau) \frac{\partial}{\partial w_k} y_k(t-\tau) \right]. \quad (6.31)$$

It should be noted that there is no change to the gradient equation for the gate, other than that the posterior probabilities are calculated using the chi-square distribution.

6.5.4 Simulation

We tested both the ME model and our ME with memory model using a simple toy time series with two regimes. The time series consisted of a switching AR(1) process driven by white noise. Each regime consisted of 200 samples and appears only once, for a total time series length of 400 samples. Each model employed two MA(1) linear predictors as experts, and used a two hidden layer MLP as the gate. While the two experts made their predictions based on the previous value only, the gate used the previous ten values, resulting in a gate architecture of 10-4-3-2.

The results for the ME model are shown in Figure 6-6, while those for our ME with memory model are shown in Figure 6-7. The solid lines represent expert I, while the dashed lines represent expert II. Note that the time scale in Figure 6-7 is ten times that of Figure 6-6, representing the fact that our gate fires only once every 10 samples.

A very simple time series was chosen to emphasize that the classical ME model makes frequent spurious switching errors even when the regimes are clearly visible to the eye, as can be seen in Figure 6-6. However, in our model, only one switching error occurs. Fur-

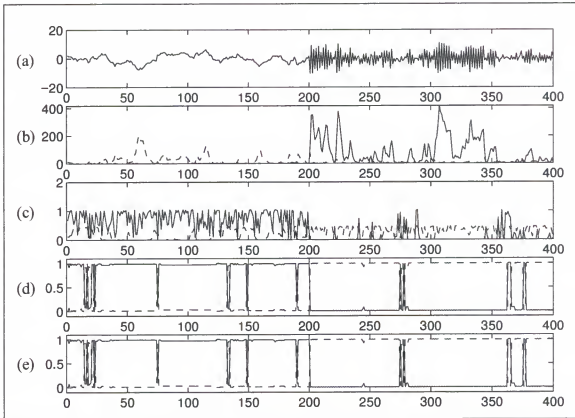


Figure 6-6. MOE simulation (Expert I (solid) and Expert II (dashed)): (a) input signal; (b) squared error; (c) pdf; (d) gate; (e) posterior probability.

thermore, we see that the transition between regimes as indicated by the gate occurs instantaneously.

It is natural to ask whether the gate output of Figure 6-6 could be “filtered and down-sampled”, in a probabilistic sense, to improve the segmentation. Such a technique is fraught with difficulties, however, because probabilities are multiplicative. A very small probability surrounded in time by unity probabilities still results in a very small probability when the events are grouped as an ensemble. Our algorithm is not subject to this, because the gate learns to produce a single output that is the result of the local performance of the experts over time.

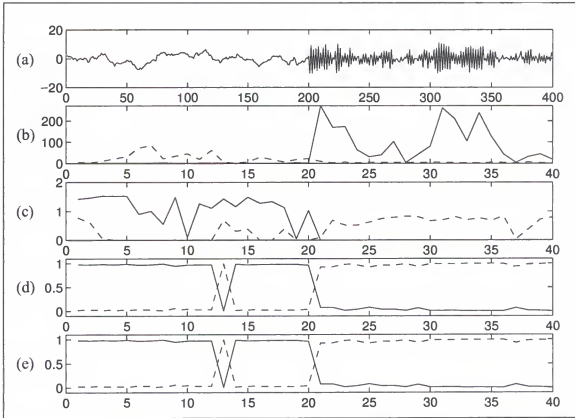


Figure 6-7. MOE with memory simulation (Expert I (solid) and Expert II (dashed)): (a) input signal; (b) squared error; (c) pdf; (d) gate; (e) posterior probability.

6.6 Adding Recursive Memory to the Mixture of Experts

Again, we model the root mean square error as in (6.22), except now we use the recursive estimate of (6.3), both repeated here for convenience

$$\chi_k(t) = \sqrt{\varepsilon_k(t)} \quad (6.32)$$

$$\varepsilon_k(n) = \lambda e_k^2(n) + (1 - \lambda)\varepsilon_k(n-1) \quad 0 < \lambda < 1. \quad (6.33)$$

6.6.1 Gamma Distribution of the Recursive Mean Square Error

Dropping reference to the particular expert momentarily, under the assumption that the instantaneous error is gaussian distributed, the exact distribution of χ is unknown. However, we propose to try and fit a gamma distribution:

$$p(\chi) = \frac{1}{K} \chi^\alpha e^{\frac{-\chi}{\beta}} \quad (6.34)$$

$$K = \frac{1}{2} b^{\frac{(a+1)}{2}} \Gamma\left[\frac{a+1}{2}\right]. \quad (6.35)$$

In order to evaluate the coefficients α and β , we take the 2nd and 4th moments of both the probability distribution and the root mean square error estimator, and equate them. From the probability distribution:

$$E[\chi^2] = \int_0^\infty \chi^2 p(\chi) d\chi = \frac{(\alpha+1)\beta}{2} \quad (6.36)$$

$$E[\chi^4] = \int_0^\infty \chi^4 p(\chi) d\chi = \frac{(\alpha+1)(\alpha+3)\beta^2}{4}. \quad (6.37)$$

We now evaluate the expectations using the recursive mean square error estimator. From (6.32),

$$E[\chi^2(t)] = E[\varepsilon(t)] \text{ and } E[\chi^4(t)] = E[\varepsilon^2(t)]. \quad (6.38)$$

From (6.33)

$$E[\varepsilon(t)] = \lambda E[e^2(t)] + (1-\lambda)E[\varepsilon(t-1)] \quad (6.39)$$

$$E[\varepsilon^2(t)] = \lambda^2 E[e^4(t)] + 2\lambda(1-\lambda)E[e^2(t)]E[\varepsilon(t-1)] + (1-\lambda)^2 E[\varepsilon^2(t-1)] \quad (6.40)$$

where e^2 is distributed according to a first order chi-square distribution, so that

$$E[e^2(t)] = \sigma^2 \quad (6.41)$$

$$E[e^4(t)] = 3\sigma^4. \quad (6.42)$$

Inserting (6.41) into (6.39) and (6.40), and (6.42) into (6.40), and simplifying:

$$E[\varepsilon(t)] = \sigma^2 \quad (6.43)$$

$$E[\varepsilon^2(t)] = \frac{2+\lambda}{2-\lambda} \sigma^4. \quad (6.44)$$

Equating (6.36) and (6.43), as well as (6.37) and (6.44),

$$\frac{(\alpha+1)\beta}{2} = \sigma^2 \quad (6.45)$$

$$\frac{(\alpha+1)(\alpha+3)\beta^2}{4} = \frac{2+\lambda}{2-\lambda} \sigma^4 \quad (6.46)$$

and solving for α and β

$$\alpha = \frac{2(1-\lambda)}{\lambda} \quad \beta = \frac{2\lambda\sigma^2}{2-\lambda}. \quad (6.47)$$

The desired probability distribution is thus

$$p(\chi) = \frac{1}{K} \chi^{\frac{2(1-\lambda)}{\lambda}} e^{\frac{-(2-\lambda)}{2\lambda\sigma^2} \chi^2} \quad (6.48)$$

$$K = \frac{1}{2} \left(\frac{\sigma^2}{\frac{1}{\lambda} - \frac{1}{2}} \right)^{\frac{1}{\lambda} - \frac{1}{2}} \Gamma \left[\frac{1}{\lambda} - \frac{1}{2} \right]. \quad (6.49)$$

In order to check the quality of the approximation, the difference between the theoretical and experimental distributions, based on filtering 100,000 points, is shown in Figure 6.8.

Having determined the parameters of the gamma distribution from the 2nd and 4th moments, it is of interest to see how the 1st moment compares to the experimental data. Figure 6-9 below show the theoretical and experimental first moment for various memory depths.

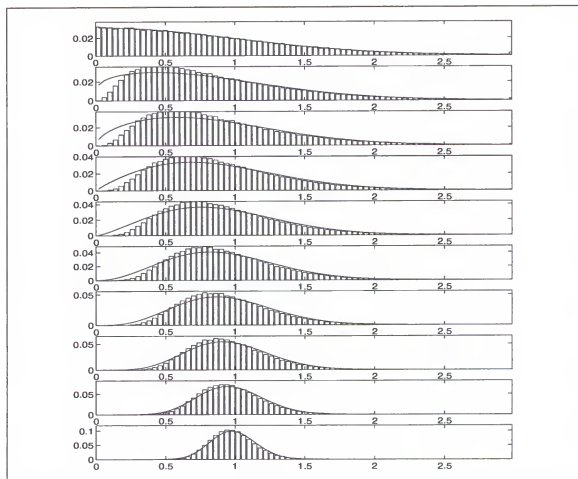


Figure 6-8. Theoretical (line) vs. experimental (histogram) χ distribution from $\lambda=1$ (top panel) to $\lambda=0.1$ (bottom panel) in increments of $1/10$.

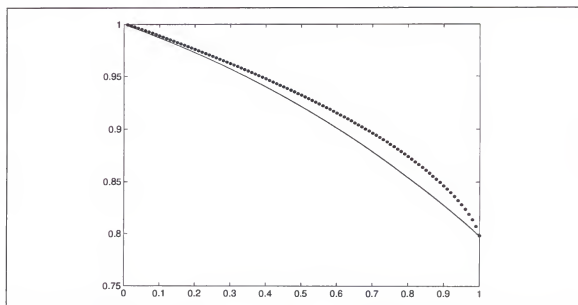


Figure 6-9. Theoretical mean (solid) and experimental mean (dotted) vs. λ .

6.6.2 Gradient Descent

We now proceed to find the gradient descent equations for the Mixture of Experts when the pdf's are given by (6.48). Once again, there is no change to the gradient equation for the gate, other than that the posterior probabilities are calculated using the gamma distribution. However, the update of the weights of the experts is different. We first calculate

$$\frac{\partial p_k}{\partial w_k} = \left[\frac{2(1-\lambda_k)}{\varepsilon_k} - \frac{(2-\lambda_k)}{\sigma_k^2} \right] p_k \lambda_k z_k \quad (6.50)$$

$$z_k(t) = \lambda_k e_k(t) \frac{\partial}{\partial w_k} y_k(t) + (1-\lambda_k) z_k(t-1) \quad (6.51)$$

Plugging (6.50) into (3.20), we obtain

$$\frac{\partial C}{\partial w_k} = \sum_{t=1}^N \left[\frac{(2-\lambda_k)}{\sigma_k^2} - \frac{2(1-\lambda_k)}{\varepsilon_k} \right] h_k z_k. \quad (6.52)$$

The equation for the variances at the end of each epoch is the same as in (6.30).

There are two ways that the algorithm reduces to the mixture of experts. When $\lambda=1$

$$\frac{\partial C}{\partial w_k} = \sum_{t=1}^N \frac{h_k e_k \partial y_k}{\sigma_k^2 \partial w_k} \quad (6.53)$$

which is identical to the normal MOE. Also, in the steady state for the winning predictor

$$\varepsilon_k(t) \approx \sigma_k^2 \quad (6.54)$$

$$\frac{\partial C}{\partial w_k} = \sum_{t=1}^N \frac{h_k z_k(t)}{\sigma_k^2} \quad (6.55)$$

which is the mixture of experts with momentum learning of the winning expert's weights.

No simulations are presented for this algorithm, as it performed poorly compared with the MOE with static memory. It was presented, nonetheless, since future applications cannot necessarily be anticipated.

CHAPTER 7 CONCLUSIONS AND FUTURE WORK

7.1 Comparison of Present Work to Classical Segmentation

We've seen how gated competitive systems can be used to segment and model piecewise stationary signals in a completely unsupervised fashion. We now compare the classical method of generalized likelihood ratio (GLR) to the new paradigm of gated competitive systems, in the context of time series segmentation. As previously stated, the GLR algorithm is a segmentation method that detects a *transition* between two adaptive models, by considering the likelihood of all possible changepoints within the current subset of data. Once a transition is detected, the algorithm is restarted anew from the changepoint. Although the GLR develops a model for the data both before and after a local changepoint, this information is usually discarded. Thus, the GLR is local in both time and model.

Gated competitive experts (GCE), on the other hand, attempt to model all the data simultaneously using a finite mixture of adaptable models. It is thus global in time and local in model. In this sense, it differs from the GLR in that a particular model (expert) can be active over non-contiguous regions of the data set. Both approaches have advantages and disadvantages, which we will now discuss.

Certainly the biggest advantage of GLR over GCE is that it segments the data in a single pass. This does not mean that the GLR is less computationally intensive. However, certain approximations to the core GLR algorithm can reduce the computational complex-

ity at a cost of increased false and/or missed detections. Because the GCE is global in time, it requires several iterations through the entire data set. However, this is also the biggest advantage of the GCE over the GLR. By being global in time, the GCE takes advantage of redundancies in the data, in the form of repeating regimes, to form better estimates of the various models that produced the regimes. However, because the GCE algorithm possesses many local minima, it is not necessarily a more reliable estimator of the change-points themselves.

When a time series includes one or more very short stationary segments, the GCE approach is clearly superior because it has a much smaller deadzone for changepoint detection than the GLR. In fact, its deadzone is basically the number of taps of the tapped delay line at the input, as long as the regime is redundant and occurs elsewhere within the time series. The deadzone of the GCE algorithm depends on the complexity of the models and the detection threshold, but it is very unreliable to detect a changepoint in a segment of data less than, say, 50 samples, since the “variance” of the autocorrelation estimate increases with decreasing samples according to a Wishart distribution.

If the time series was produced by piecewise non-linear dynamical systems, then the GCE approach is clearly superior to the GLR algorithm. The GLR algorithm becomes intractable when the models are non-linear, because the models must be trained to completion for each potential changepoint within each block. The GCE experts, even if non-linear, need not be trained to completion at each iteration for the algorithm to converge to a valid solution. Also, the GCE algorithm lends itself to parallelization, since each expert can be run on a separate machine.

We now ask the question whether the GLR algorithm could be adapted so that it too makes use of non-contiguous regions that may represent the same dynamical system. Certainly, the model parameters of each stationary region encountered can be stored instead of discarded. However, in order to globally model the data similar to the gated competitive systems approach, the number of generated models must be limited. Thus, once a new changepoint (and hence stationary region) is detected, some decision would have to be made as to whether the current model is sufficiently “close” to one of the stored models. If the current model is a sufficient match for a stored model, then the stored model’s parameters are updated to reflect the addition of the new data. If the current model is sufficiently different from all stored models, then a new reference model is created and stored. For this approach to work, there must be a metric for measuring the difference between the reference model and the current model. For linear models, there are several well known distance metrics. If the models are non-linear, it is much more difficult to define a distance metric. In addition, it may not be possible to update the model parameters without re-access to all prior regions of data that were assigned to the model.

7.2 Comparison of Input and Output Based Competitive Systems

Recall that an input based gating system, such as the Mixture of Experts, uses a neural network that learns to mediate which expert is valid based only on the input data. On the other hand, output based gating systems, such as the Annealed Competition of Experts, the Neighborhood Map of Competing Predictors, and Self-Annealing Competitive Prediction, continually reassess which expert is valid based on their recent performance.

There is a significant difference between input and output based gating when the experts are predictors. Because output based gating always requires a performance mea-

sure to assign validity to the experts, it cannot be used for multi-step prediction. This means that it is restricted to use as an analysis tool. However, this is usually not a serious restriction because multi-step prediction requires a higher level modeling of the transitions between stationary regimes. If an input based gate system is performing multi-step prediction and makes an error in expert assignment, the prediction will rapidly diverge.

Higher level modeling of the transitions between stationary regimes is an area of research that has only just begun to be explored, and could eventually lead to speech recognition systems that completely self-organize around a few minutes of spoken text.

7.3 Competitive PCA

Competitive PCA is a valid alternative to competitive prediction for the segmentation problem. The segmentation can be provided either by the gate or the posterior probabilities. Recall that the gate is a neural network whose input is the original signal and whose i^{th} output is an estimate of the prior probability of the i^{th} PCA expert. The posterior probability of the i^{th} PCA expert is the gate output multiplied by the pdf of the i^{th} PCA expert evaluated at the local reconstruction error. During training, the posterior probabilities act as targets for the gate. However, the role of the gate is not to merely memorize the posterior probabilities, but rather to smooth them. Once trained, *the output of the gate alone is capable of segmenting a new signal*. Unfortunately, because the gate is a neural network, generalization performance can vary widely depending on the number of adaptable parameters, the quality of the training run, etc. However, even if the gate is not generalizing well, the posterior probabilities can be calculated, and these used to segment the signal, at the cost of a noisier segmentation.

7.3.1 Application to Images

For images, two-dimensional prediction is not well-defined, and thus PCA becomes a natural analysis tool. In chapter four, we saw that competitive PCA can be used to segment textures within an image in a completely unsupervised fashion. The technique is limited in a practical sense to textures that are defined over short distance scales. Also, autocorrelation is known to be an incomplete description of textures.

PCA is sensitive to the *orientation* of textures with directional energy. Thus, while humans naturally recognize rotations of a texture as belonging to the same class, such rotations equally affect the PCA representation (masks) and cause performance degradation when analyzing new test images with rotated textures. One way around this would be to re-train the gate, after the basic algorithm has converged, with rotated versions (90, 180, and 270 degrees) of the training image, keeping the desired response the same.

7.3.2 Application to Time Series

When the system input is delayed versions of a time series, the same competitive PCA system can be used to segment time signals. We saw how temporal PCA picks out peaks in the signal spectrum as a matched filter. This makes temporal PCA ideally suited to segmenting signals dominated by a few strong linear modes. Temporal PCA also is the optimal signal to noise projection for additive white noise and the optimal linear encoder.

7.4 Temporal PCA and the Discrete Prolate Spheroids

We saw how the discrete prolate spheroids are a natural basis for the eigenvectors of the autocorrelation matrix of a time series (temporal PCA). Intuition says that it should be possible to perform faster on-line adaptation of the principal components using the discrete prolate spheroid sequences (DPSS) as a basis. Only two parameters would need to be

adapted for the first and largest eigenvectors: the center frequency and amplitude of the DPSS. However, the basis is still complete for all center frequencies, and thus gradient descent with respect to that parameter cannot be used. Some other criteria, such as the Frobenius norm of the expansion coefficients is required.

APPENDIX PROPERTIES OF THE DISCRETE PROLATE SPHEROIDS

We now present some of the properties of Slepian's (1978) discrete prolate spheroid sequences (DPSS) and wave functions (DPSWF). In the time domain, the index limited DPSS are the eigenvectors of

$$\mathbf{R} \mathbf{v}_k = \Lambda_k \mathbf{v}_k \quad k = 0 \dots N-1 \quad (\text{A.1})$$

when the components of \mathbf{R} are given by the sinc function

$$R_{mn} = r(m-n) = \sin[2\pi W(m-n)]/(m-n). \quad (\text{A.2})$$

The index limited DPSS are alternately symmetric and anti-symmetric

$$\begin{aligned} v_k(n) &= v_k(N-1-n) & k &= \text{even} \\ &= -v_k(N-1-n) & k &= \text{odd} \end{aligned} \quad (\text{A.3})$$

The DTFT of the eigenvalue equation (A.1) defines the discrete prolate spheroidal wave functions (DPSWF)

$$\int_{-W}^W \frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} U_k(f', N, W) df' = \Lambda_k(N, W) U_k(f, N, W) \quad (\text{A.4})$$

where the DPSWF are related to the index limited DPSS through the DTFT

$$U_k(f) = \epsilon_k \sum_{n=0}^{N-1} v_k(n) e^{i2\pi f[n-(N-1)/2]} \quad (\text{A.5})$$

and where ϵ_k is a constant (1 or i) defined to ensure that the DPSWF are real. The DPSWF are doubly orthogonal

$$\frac{1}{\Lambda_n(N, W)} \int_{-W}^W U_m(f, N, W) U_n(f, N, W) df = \int_{-1/2}^{1/2} U_m(f, N, W) U_n(f, N, W) df = \delta_{mn}. \quad (\text{A.6})$$

Because of this double orthogonality property, the inverse DTFT is expressible over two ranges

$$\begin{aligned} v_k(n) &= \frac{1}{\varepsilon_k} \int_{-1/2}^{1/2} U_k(f, N, W) e^{-i2\pi f[n-(N-1)/2]} df \\ &= \frac{1}{\varepsilon_k \Lambda_k(N, W)} \int_{-W}^W U_k(f, N, W) e^{-i2\pi f[n-(N-1)/2]} df \end{aligned} \quad (\text{A.7})$$

The Dirichlet kernel is expressible in terms of the DPSWF

$$\frac{\sin[N\pi(f-f')]}{\sin[\pi(f-f')]} = \sum_{n=0}^{N-1} U_n(f-f_0, N, W) U_n(f'-f_0, N, W). \quad (\text{A.8})$$

where f_0 is an arbitrary frequency offset.

Even though there is no general analytical expression for the DPSWF, the DPSS and DPSWF have a number of interesting properties. Of all index limited sequences of length N , the index limited DPSS have the greatest concentration of energy in the band $|f| \leq W$, subject to orthogonality constraints. The eigenvalues represent the fraction of energy within the band. Since the DPSS and DPSWF are still subject to the so called time bandwidth product law, the first $2NW$ eigenvalues are close to one, with the remaining eigenvalues approaching zero.

If $U_{N-1}(f, N, W)$ has the smallest possible energy concentration within $|f| \leq W$, then it has the largest for $|f| > W$, and thus $U_{N-1}(\frac{1}{2}f, N, \frac{1}{2}W) = U_0(f, N, W)$. This generalizes to all pairs of DPSWF's as follows

$$U_k(f, N, W) = U_{N-1-k}\left(\frac{1}{2}-f, N, \frac{1}{2}-W\right). \quad (\text{A.9})$$

As $W \rightarrow 0$, the k^{th} DPSWF evaluated at fW becomes the k^{th} Legendre polynomial. Using (A.9), one can also show that as $W \rightarrow \frac{1}{2}$, the DPSWF evaluated at fW are the Legendre polynomials.

The following figures demonstrate some of the properties of the DPSWF for the case when $N=7$, the largest value for which the software program Mathematica v3.0 can find a symbolic solution to the time domain eigenvalue equation.

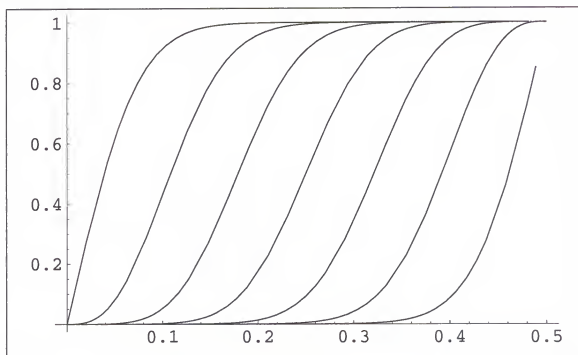


Figure A-1. Eigenvalues $\Lambda_k(7, W)$ versus half-bandwidth, W , for $k=0 \dots 7$.

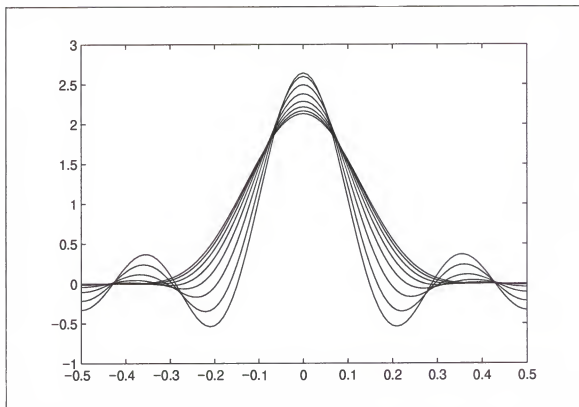


Figure A-2. $U_0(f, 7, W)$ versus f for $W=0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35$, and 0.4 .

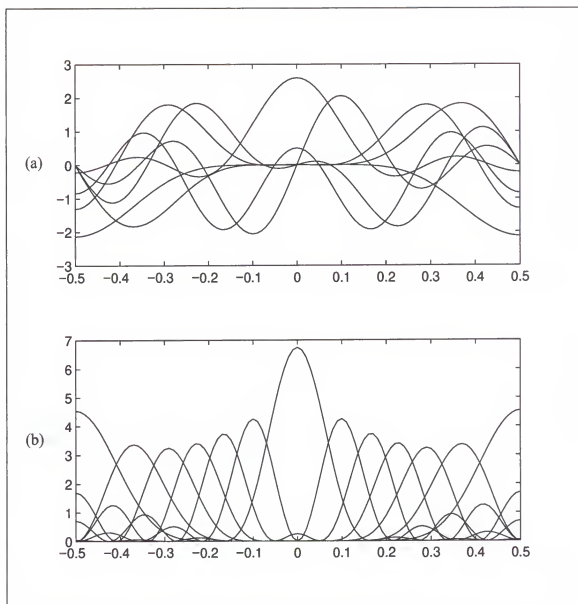


Figure A-3. (a) $U_k(f, 7, 0.1)$ and (b) $U_k^2(f, 7, 0.1)$ versus f for $k=0 \dots 7$

REFERENCES

Andersson P., Adaptive forgetting in recursive identification through multiple models, *Int. J. Control*, vol. 42, no. 5, pp. 1175-1193, 1985.

Andre-Obrecht R., A new statistical approach for the automatic segmentation of continuous speech signals, *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 36, no. 1, pp. 29-40, 1988.

Appel U. and Brandt A.V., Adaptive sequential segmentation of piecewise stationary time series, *Inf. Sci.*, vol. 29, no. 1, pp. 27-56, 1982.

Arbib M. (Ed.), *The Handbook of Brain Theory and Neural Networks*, MIT Press, Cambridge MA.

Arfken G., *Mathematical Methods for Physicists*, Academic Press, Orlando, 1985.

Basseville M., Detecting Changes in Signals and Systems - A Survey, *Automatica*, vol. 24, no. 3, pp. 309-326, 1988.

Basseville M. and Benveniste A., Sequential detection of abrupt changes in spectral characteristics of digital signals, *IEEE Trans. on Information Theory*, vol. IT-29, no. 5, 1983.

Basseville M. and Nikiforov I.V., *Detection of Abrupt Changes, Theory and Application*, Prentice-Hall, Englewood Cliffs, 1993.

Bishop C.M., *Neural Networks for Pattern Recognition*, Clarendon Press, Oxford, 1995.

Box G.E.P. and Jenkins G.M., *Time Series Analysis: Forecasting and Control*, revised edition, Holden-Day, San Francisco, CA, 1976.

Brandt A.V., Detecting and estimating parameter jumps using ladder algorithms and likelihood ratio tests, *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1017-1020, 1983.

Brillinger D., *Time Series: Data Analysis and Theory*, Holden-Day, San Francisco, 1975.

Broomhead D.S. and King G.P., Extracting qualitative dynamics from experimental data, *Physica D*, vol. 20, no. 2-3, pp. 217-236, 1986.

- Casdagli M., Eubank S., Farmer J.D., and Gibson J., State space reconstruction in the presence of noise, *Physica D*, vol. 51, pp. 52-98, 1991.
- Chu C., Time series segmentation: A sliding window approach, *Information Sciences*, vol. 85, pp. 147-173, 1995.
- Danilov D., Solnsev V., and Zhigljavsky A., Analysis and forecast of time series on the base of principal components, *Second Int. Conf. on Computing in Economics and Finance*, Geneva, Switzerland, 1996.
- Dempster A.P., Laird N.M., and Rubin D.B., Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B*, 39, 1-38, 1972.
- Dony R. and Haykin S., Optimally adaptive transform coding, *IEEE Trans. on Image Processing*, vol. 4, no. 10, 1995.
- Doutriaux A. and Zipser D., Unsupervised discovery of speech segments using recurrent nets, *Proc. of the 1990 Connectionist Models Summer School*, pp. 303-309, 1990.
- Elman J.L., Finding structure in time, *Cognitive Science*, vol. 14, pp. 179-211, 1990.
- Fancourt C. and Principe J., Competitive principal component analysis for locally stationary time series, *IEEE Trans. on Signal Processing*, vol. 46, no. 11, pp. 3068-3081, 1998a.
- Fancourt C. and Principe J., Modeling time dependencies in the mixture of experts, *Proc. of the IEEE Int. Joint Conf. on Neural Networks (IJCNN)*, vol. 3, pp. 2324-2327, 1998b.
- Fancourt C. and Principe J., Temporal self-organization through competitive prediction, *Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, pp. 3325-3328, 1997a.
- Fancourt C. and Principe J., Soft competitive principal component analysis using the mixture of experts, *Image Understanding Workshop*, vol. 2, pp. 1071-1075, 1997b.
- Fancourt C. and Principe J., A neighborhood map of competing one step predictors for piecewise segmentation and identification of time series, *IEEE Int. Conf. on Neural Networks (ICNN)*, vol. 4, pp. 1906-1911, 1996a.
- Fancourt C. and Principe J., Temporal self-organization through competitive prediction, *Proceedings of the Ninth Yale Workshop on Adaptive and Learning Systems*, pp. 8-12, 1996b.
- Fukunaga K., *Statistical Pattern Recognition*, Academic Press, 1990.

Gibson J. F., Farmer J.D., Casdagli M., and Eubank S., *An Analytic Approach to Practical State Space Reconstruction*, Santa Fe Institute Report, 92-04-021, 1992.

Haykin S., *Neural Networks: A Comprehensive Foundation*, New York: Macmillan, 1994.

Hertz J., Krogh A., and Palmer R., *Introduction to the Theory of Neural Computation*, Addison Wesley, Redwood City, 1991.

Hecht-Nielsen R., Nearest matched filter classification of spatiotemporal patterns, *Applied Optics*, vol. 26, no. 10, pp. 1892-1899, 1987.

Jacobs, R. A., Bayesian approach to model selection in hierarchical mixtures-of-experts architectures, *Neural Networks*, vol. 10, no.2, pp. 231-241, 1997.

Jacobs R.A., Jordan M.I., Nowlan S.J., and Hinton G.E., Adaptive mixtures of local experts, *Neural Computation*, vol. 3, pp. 79-87, 1991.

Jolliffe I., *Principal Component Analysis*, Springer-Verlag, New York, 1986.

Jordan M.I. and Jacobs R.A., Hierarchical mixtures of experts and the EM algorithm, *Neural Computation*, vol. 6, pp. 181-214, 1994.

Jordan M.I. and Xu L., Convergence results for the EM approach to mixtures of experts architectures, *M.I.T. Artificial Intelligence Lab*, A.I. Memo No. 1458, 1993.

Kadirkamanathan V., Recursive nonlinear identification using multiple model algorithm, *Neural Networks for Signal Processing*, pp. 171-180, 1995.

Kay S., *Modern Spectral Estimation*, Prentice Hall, 1988.

Kligiene N. and Telksnys L., Methods of detecting instants of change of random process properites, *Automation and Remote Control*, vol. 44, no. 10, Part II, pp. 1241-1283, 1983.

Kohlmorgen J., Muller K.-R., and Pawelzik K., Improving short-term prediction with competing experts, *ICANN 95: Proc. of the Int. Conf. on Artificial Neural Networks*, EC2 & Cie, Paris, 2:215-220, 1995

Kohonen T., Self-organized formation of topologically correct feature maps, *Biological Cybernetics*, vol. 43, pp. 59-69, 1982.

Kohonen T., The self-organizing map, *Proc. IEEE*, vol. 78, no. 9, pp. 1464-1480, 1990.

Lakkis I. and McLernon D., Optimum eigenfilters and matched filters, *Electronics Letters*, vol. 32, no. 22, pp. 2068-2070, 1996.

Lawley D.N., A modified method of estimation in factor analysis and some large sample results, *Uppsala Symposium on Psychological Factor Analysis*, pp. 35-42, 1953.

Levin E., Modeling time varying systems using hidden control neural architecture, *Advances in Neural Information Processing Systems* 3, pp. 147-154, 1990.

Makhoul J., On the eigenvectors of symmetric Toeplitz matrices, *IEEE Tran. on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, pp. 868-872, 1981.

Michalopoulou Z.H., Nolte L.W., and Alexandrou D., Performance evaluation of multilayer perceptrons in signal detection and classification, *IEEE Trans. on Neural Networks*, vol. 6, no. 2, pp. 381-386, 1995.

Morrison D., *Multivariate Statistical Methods*, McGraw-Hill, 1976.

Muller K.R., Kohlmorgen J., and Pawelzik K., Analysis of switching dynamics with competing neural networks, *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E78-A, no. 10, pp. 1306-1315, 1995.

Niedzwiecki M., Identification of nonstationary stochastic systems using parallel estimation schemes, *IEEE Trans. on Automatic Control*, vol. 35, no. 3, pp. 329-334, 1990.

Niedzwiecki M., Multiple model approach to finite memory adaptive filtering, *IEEE Trans. on Signal Processing*, vol. 40, no. 2, pp. 470-473, 1992.

Niedzwiecki M., Identification of time-varying systems with abrupt parameter changes, *Automatica*, vol. 30, no. 3, pp. 447-459, 1994.

Page E.S., Estimating the point of change in a continuous process, *Biometrika*, vol. 44, pp. 248-252, 1957.

Pawelzik K., Kohlmorgen J., and Muller K.R., Annealed competition of experts for a segmentation and classification of switching dynamics, *Neural Computation*, 8, pp. 340-356, 1996.

Peebles, P.Z., *Probability, Random Variables, and Random Principles*, McGraw-Hill, New York, 1993.

Pisarenko V.F., The retrieval of harmonics from a covariance function, *Geophys. J. Royal Astron. Soc.*, pp. 347-366, 1973.

Priestley M.B., *Non-linear and non-stationary time series analysis*, Academic Press, San Diego, 1988.

- Principe J., deVries B., and Guedes de Oliveira P. The gamma filter: a new class of adaptive IIR filters with restricted feedback. *IEEE Trans. Signal Processing*, 41(2):649-656, 1993.
- Rabiner L. and Juang B.H., *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, 1993.
- Sanger T., Optimal unsupervised learning in a single-layer linear feedforward neural network, *Neural Networks*, vol. 2, pp. 459-473, 1989.
- Slepian D., Prolate spheroidal wave functions, Fourier analysis, and uncertainty-V: the discrete case, *Bell System Technical Journal*, vol. 57, no. 5, 1978.
- Srivastava A.N. and Weigend A.S., Improved time series segmentation using gated experts with simulated annealing, *Proc. of the 1996 IEEE Int. Conf. on Neural Networks (ICNN)*, vol. 4, pp. 1883-1888, 1996.
- Thomson D.J., Spectrum estimation and harmonic analysis, *Proc. of the IEEE*, vol. 70, no. 9, pp. 1055-1096, 1982.
- Tong H. and Lim K.S., Threshold autoregression, limit cycles, and cyclical data, *J. Roy. Stat. Soc. B*, vol. 42, pp. 245-292.
- Van Trees H., *Detection, Estimation and Modulation Theory: Part I*, Wiley, New York, 1968.
- Waterhouse S.R and Robinson A.J., Pruning and growing hierarchical mixtures of experts, *Proc. of the 4th Int. Conf. on Artificial Neural Networks*, n 409, pp. 341-346, 1995.
- Weigend A.S., Mangeas M., and Srivastava A.N., Nonlinear gated experts for time series: discovering regimes and avoiding overfitting, *International Journal of Neural Systems*, vol. 6, no. 4, 1995a.
- Weigend A.S. and Srivastava A.N., Predicting conditional probability distributions: A connectionist approach, *International Journal of Neural Systems*, vol. 6, pp. 109-118, 1995b.
- Widrow B. and Stearns S.D., *Adaptive Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1985.
- Xu L., Signal segmentation by finite mixture model and EM algorithm, *Proc. Int. Symp. Artif. Neural Nets, ISANN'94*, pp. 453-458, 1994.

BIOGRAPHICAL SKETCH

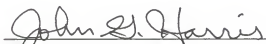
Craig Langdale Fancourt was born in Trenton, New Jersey, on October 25, 1960. He attended Council Rock High School in Newton, Pennsylvania, from 1974 through 1977. In 1977, at the age of 16, he began undergraduate studies at the University of South Florida in Tampa, Florida, graduating with a B.S. in physics in 1982. From 1982 through 1988, he was an Engineer of Measurements and Standards at the General Electric Neutron Devices Dept. in Largo, Florida, where he was responsible for the engineering of all electrical, mechanical, and radiative calibrations of all plant instrumentation. Taking a leave of absence from his position at G.E., Mr. Fancourt returned full-time to the University of South Florida for the 1986-87 academic year, where he taught 3 undergraduate physics labs per semester and received his M.S. in physics in 1987. He then returned to G.E., where he worked until 1988. From 1988 through 1991, Mr. Fancourt was Section Head of the electrical calibration lab, part of the Measurement Standards Lab, at the Research Institute of King Fahd University of Petroleum and Minerals in Dhahran, Saudi Arabia. There, he oversaw all electrical calibrations both in support of university research and for customers throughout the Middle East. In 1992, he began graduate studies in digital signal processing in the Electrical Engineering Dept. at the University of Florida in Gainesville, Florida. From 1994 through 1998, he was a Research Assistant in the Computational Neuro-Engineering Lab, except for a one-year assignment as an algorithm programmer at NeuroDimension, Inc.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.




José C. Principe, Chair
Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



John G. Harris
Assistant Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Jian Li
Associate Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Jacob Hammer
Professor of Electrical and Computer Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Howard B. Rothman
Professor of Communication Sciences and Disorders

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

December 1998

A handwritten signature in dark ink, appearing to read 'W. M. Phillips', is written over a horizontal line.

Winfred M. Phillips
Dean, College of Engineering

M. J. Ohanian
Dean, Graduate School